**Appendix A**

# MDA Software Applications Summary Table

Wireless Medical Enterprise Working Group http://www.tatrc.org

Telemedicine & Advanced Technology Research Center (TATRC) http://www.tatrc.org

US Army Medical Research & Materiel Command (USAMRMC Fort Detrick, MD)

**Send information on additonal applications to: TATRC WME** **Note: Last edit by DM 1/22/02 1330 (edit each revi

| m | Application Name/Title | Type of Application | Description | Palm OS | WinCE/ PC |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| 4T Medical 1.3 | Patient Charting software | Database application for tracking personal medical information | Yes | NA |
| ABGPro | Medical Calculator | Arterial Blood Gas Interpretation | Yes | NA |
| Adobe Acrobat Reader for the Palm/Pocket PC | Adobe Acrobat reader | view Adobe Acrobat files | Yes | NA |
| Antibiotic Guide | Medical Reference | Antibiotic Guide from Johns Hopkins University | Yes | Ye |
| AportisDoc Reader | doc. Reader | Doc reader | Yes | NA |
| Body Mass Index Calculator | Medical Reference | Calculates BMI using English Metric System | NA | NA |
| Cardio Textbook | Medical Reference | Cardiac problem reference (iSilo Format) | Yes | NA |
| CardioFILE 1 0 0 | Patient Charting software | For a Cardiologist to keep track of patients, Clinical Files, Progress Notes and Appointments | Yes | NA |
| Complete Surgical Knots | Medical Reference | What suture and what type of sutures to use on a wound | Yes | NA |
| Creatinine Clearance Calc 1 3 | Medical Calc | Calculate Creatinine clearance. How good their kidneys work | Yes | NA |
| CSpotRun | doc. Reader | Doc reader | Yes | NA |

| | | | | |
|---|---|---|---|---|
| Current Medical Therapeutics 1 0 | Medical Reference | Up to date quidelines and recommendations for treatment for common acute medical conditions | Yes | NA |
| Deployment Vaecines and Medications | Medical Reference | From the CHPPM Website | Yes | NA |
| Digital Assist | Patient Charting software | Patient Charting Software | Yes | NA |
| Docucare | Patient Charting software | Patient Charting Software | Yes | NA |
| Electrolytes First | Medical Reference | Chapters of metabolism: ecttolytes hypo and hyper .kaliemia, natremia, etc | Yes | NA |
| ePhysician | Medical Reference | Drug reference, Monitoring and absorption information,Lab test interferences | Yes | NA |
| ePocrates | Pharmacy, Medical Reference | Provides up to date information on drugs and drug interactions, dosage levels,Pregnancy/Lactation | Yes | NA |
| Ferri's Practical Guide to the Care of the Medical Patient | Medical Reference | A clear and concise reference for the busy clinician, that provides the very latest in clinical information, drug therapeutics and laboratory medicine | Yes | Yes |
| Gastroenterology | Medical Reference | for rotations in surgery or in internal medicine | Yes | NA |
| Global Expeditionary Medical System (GEMS) | Patient Charting software | Medical Surveillance System | Yes | Yes |
| Griffith's 5-Minute Clinical Consult | Medical Reference | An index that is organized by Diagnosis, Treatment, Medications and Follow-ups | Yes | Yes |
| HanDBase | Relational Database | Relational Database | Yes | Yes |

| | | | | |
|---|---|---|---|---|
| Harrison's Principles of Internal Medicine 14e Companion | Medical Reference | Summarization of the diagnosis and treatment of all disorders commonly seen in the clinical | Yes | NA |
| HealthEngage Asthma | Patient Charting software | tracking your peak flow, medications, and observations, or to manage your asthma | Yes | NA |
| Hospital Corpsman Sickcall Screeners Handbook | | Program is directed at junior Hospital Corpsman (E-2 to E-4) | Yes | NA |
| Hospital Rounds Database | Patient Charting software | Organize clinical, billing and coding information | Yes | NA |
| Immunization Guide 2.0 | Medical Reference | 2000 American Academy of Pediatrics immunization schedule Detailed information on vaccines | Yes | NA |
| Internal Medicine Notes 1.2 | Medical Reference | Doc formatted Internal Medicine and Bookmarks | Yes | NA |
| iScribe | Medical Reference | Drug reference, Monitoring and absorption information,Lab test interferences | Yes | Yes |
| IV Rate Calc 1.1 | Medical Calc | Calculate an intravenous drip rate in ml/hr | Yes | NA |
| Johns Hopkins Antibiotic Guide | | Guide to a variety of handheld devices | Yes | Yes |
| LexiDrugs | Pharmacy, Medical Reference | Drug information resource guide | Yes | Yes |
| ListPro | Patient Charting software | Patient tracking, procedure logs, medical reference databases and much more | NA | Yes |
| LogonHealth | Pharmacy | wireless handheld solution allows doctors to electronically prescriptions and send them to the patient's pharmacy | Yes | Yes |
| MC4 First Responder | Patient Charting software | Patient Charting Software | Yes | NA |

| | | | | |
|---|---|---|---|---|
| MedCalc 3.2 | Medical Calc | Medical Calculator | Yes | NA |
| MedRules | Medical Calculator | Collection of common calculations/prediction tools in medicine | Yes | NA |
| MegaDoc | doc Reader | Doc reader | Yes | NA |
| Mobile Linkdoc | doc. Reader | Doc reader | Yes | NA |
| MobilePractice | Medical Reference | Provides patient management, medical reference, and current medical literature | Yes | NA |
| NAEPP Asthma Treatment Guidelines | Medical Reference | Provides succinct point-of-care asthma guidelines reference | Yes | NA |
| OEI Obesity Treatment Guidelines | Medical Reference | An interactive decision-support tool for implementing the OEI Guidelines | NA | NA |
| Ophthalmology Notes 1.11 | Medical Reference | Complete Ophthalmology Notes with different sources | Yes | NA |
| Orthopedic Surgery Notes 1.12 | Medical Reference | Complete ortho notes with pix included (iSilo Format) | Yes | NA |
| PanaRead | doc. Reader | Doc reader | Yes | NA |
| ParkStone | Medical Reference | medication management, referral generation on a network of palm-sized PCs (PPC) | NA | Yes |
| Patient Keeper | Patient Charting software | Patient Charting Software | Yes | NA |
| Patient Tracker 5.0 | Patient Charting software | Patient Charting Software | Yes | Yes |

| | | | | |
|---|---|---|---|---|
| PDA Ortho Bundle 1.0 | Medical Reference | Combines 3 Ortho documents(Ortho surgery,Ophthalomology,Surgical knots) | Yes | N/ |
| Peanunt Reader | doc. Reader | Doc reader | Yes | N/ |
| Peds Doser | Medical Calculator | Pediatric medication calculator | Yes | N/ |
| PICU 3.0 | Medical Calc | calculates various drugs that are most commonly used in pediatric I.C.U | Yes | N/ |
| PocketChart | Patient Charting software | Patient Charting Software | NA | Yc |
| PocketPhoto | Support | Everything you need to put pictures on your Palm | Yes | N/ |
| PocketScript | Pharmacy, Medical Reference | obtain patient information, insurance coverage information, formulary listings and potential drug interactions | NA | N/ |
| Preg Dates | Medical Calculator | Pregnancy Wheel for determination of EDC | Yes | N/ |
| PregTrak 3.1 | Medical Calc | Program for tracking all your OB patients. | Yes | N/ |
| QED | doc. Reader | Doc reader | Yes | N/ |
| QuickOffice | Microsoft Office reader | Work with your Microsoft Office documents whenever you need to on your Palm! | Yes | Yc |
| QVADIS Express Reader OT | doc. Reader | Doc reader | Yes | N/ |
| Raphael 99 | Patient Charting software | Patient Charting Software | Yes | Yc |
| ReadyScript | Office Management/Coding/billing | real-time prescribing and patient management functions | NA | Yc |
| Redi-Reference Inc. | Pharmacy, Medical Reference | Provides handbooks in cardiology, obstetrics and clinical guidelines | Yes | N/ |
| Rich Reader | doc. Reader | Doc reader | Yes | N/ |
| Shots 2.2 | Medical Reference | Harmonized Immunization schedule | Yes | N/ |
| SmartDoc | doc. Reader | Doc reader | Yes | N/ |

| | | | | |
|---|---|---|---|---|
| Tarascon Pharmacopeia | Pharmacy, Medical Reference | Provides up to date information on drugs and drug interactions, dosage levels,Pregnancy/Lactation | Yes | N/ |
| TealDoc | doc. Reader | Doc reader | Yes | N/ |
| TestTrakker | Diagnostic Tracker | TestTrakker™ for the Palm OS®, gives healthcare providers a simple, rapid, and intuitive risk management tool to assist them in tracking the countless numbers of laboratory and imaging tests that they order for their patients on a daily basis | Yes | N/ |
| The 1998 Guidelines for the Treatment of Sexually Transmitted Diseases | Medical Reference | STD reference | Yes | Ye |
| The CDC "Yellow Book" for Traveler's Illnesses | Medical Reference | Travelers Illneses reference | Yes | Ye |
| The Harriet Lane Handbook | Medical Reference | This authoritative reference has been in use over the last 50 years for fast, and accurate treatment of a wide variety of pediatric patients. | Yes | Ye |
| The Medical Algorithms Project | Medical Reference | Look-up table for healthcare providers | NA | NA |
| The Redi-Reference handbooks | Pharmacy, Medical Reference | Quarterly updates for one year for current content for the e-book | Yes | NA |
| Treatment of Biological Warfare Agent Casualties | Medical Reference | Treatment reference for Biological Warfare | Yes | NA |
| US Naval Hospital Flight Surgeons Handbook | Manual, Reference | Naval Aerospace Medicine and a guide for dealing with the other complex administrative procedures known as "the system " | Yes | NA |
| WardWatch | Patient Charting software | To aid medical staff involved in ward rounds | Yes | NA |
| Washington Manual of Medical Therapeutics | Medical Reference | Physician Resource textbook that has been put in palm format | Yes | NA |

| | | | | |
|---|---|---|---|---|
| Wireless MD | Pharmacy, Medical Reference | prescription writing, charge capture, physician directory, lab results, patient record information, e-mail, fax capabilities, and text-to-text messaging. | NA | NA |
| ZapCode 2.3 | Office Management/Coding/billing | Determines the Proper HCFA Evaluation and Management Code | Yes | Yc |
| Zdoc | doc. Reader | Doc reader | Yes | NA |

# SOFTWARE APPENDIX

Including
Part I, pages 1 to 62, and
Part II, pages 1 to 32.

```
1  #
2  # PIDS_globals.cpk
3  #
4  ##############################################################################
5  #
6  # PIDS (tm)
7  # Pharmacy Intervention Documentation System
8  # Copyright © 2002  Integrated Software Systems, L.L.C.
9  #
10 ##############################################################################
11 #
12 # Version:  1.0
13 # Date:     8/15/2001
14 # Author:   Todd Viegut & Brad Tice
15 #
16 #
17 # Miscellaneous functions.
18 #
19 # GLOBAL variable definitions used throughout the PIDS application.
20 #
21 variables;
22     #
23     numeric nInterventionKeyId;
24     numeric fUpdateMode = false;
25
26     #
27     numeric nLabValuesCreated;
28
29     #
30     numeric nTotalFrames = 14;  # ADJUST THIS IF ANOTHER FRAME IS ADDED TO THE ARRAY BELOW
       !!!
31     object objPIDSFrames[nTotalFrames] = PIDS_intro_frame, intervention_start_info_frame,
       drug_related_problem_frame, prescription_type_frame, patient_risk_frame,
       problem_type_frame, intervention_recommendation_frame, intervention_activity_frame,
       intervention_result_frame, intervention_time_frame, patient_info_frame,
       lab_values_frame, soap_follow_up_frame, save_intervention_frame;
32     numeric nContainsSelector[nTotalFrames] = false, false, true, true, true, true, true,
       true, true, true, false, false, false;
33     object objInterventionSelectors[nTotalFrames] = drug_related_problem_selector,
       drug_related_problem_selector, drug_related_problem_selector, rx_selector,
       patient_risk_selector, problem_type_selector, intervention_recommendation_selector,
       intervention_activity_selector, intervention_result_selector,
       intervention_time_selector, intervention_time_selector, intervention_time_selector,
       intervention_time_selector, intervention_time_selector;
34     numeric DRP_FRAME_NUMBER = 2;
35
36     # drug related problem arrays...
37     numeric nDRPSelected = 0;
38     numeric nTotalDRPFrames = 7;    # ADJUST THIS IF ANOTHER FRAME IS ADDED TO THE ARRAY
       BELOW !!!
39     object objDRPFrames[nTotalDRPFrames] = unnecessary_drug_therapy_frame,
       wrong_drug_frame, dosage_too_low_frame, adverse_drug_reaction_frame,
       dosage_too_high_frame, inappropriate_compliance_frame, need_therapy_frame;
40     object objDRPSelectors[nTotalDRPFrames] = unnecessary_drug_therapy_selector,
       wrong_drug_selector, dosage_too_low_selector, adverse_drug_reaction_selector,
       dosage_too_high_selector, inappropriate_compliance_selector, need_therapy_selector;
41     # NOTE here that element 0 is filler so that the array can be used as 1-based array...
42     numeric nDRPSelectionListOffsets[nTotalDRPFrames + 1] = 0, 0, 5, 10, 16, 22, 26, 32;
43
44     #
45     numeric nTotalTextEntries = 9;  # ADJUST THIS IF ANOTHER BUTTON IS ADDED TO THE ARRAY
       BELOW !!!
46     object objTextEntryButtons[nTotalTextEntries] = allergies_button, med_history_button,
       family_history_button, social_history_button, subjective_button, objective_button,
       assessment_button, plan_button, follow_up_button;
47     object objSearchTextEntryButtons[nTotalTextEntries] = search_allergies_button,
       search_med_history_button, search_family_history_button, search_social_history_button,
       search_subjective_button, search_objective_button, search_assessment_button,
       search_plan_button, search_follow_up_button;
48     object objTextEntryTextFields[nTotalTextEntries] = txAllergies, txMedHistory,
       txFamilyHistory, txSocialHistory, txSubjective, txObjective, txAssessment, txPlan,
```

```
        txFollowUp;
49      string sTitles[nTotalTextEntries] = "Allergies", "Med History", "Family History",
        "Social History", "Subjective", "Objective", "Assessment", "Plan", "Follow Up";
50      string sDBTextEntryMemos[nTotalTextEntries] = "", "", "", "", "", "", "", "", "";
51
52      numeric nAllergies = 0;
53      numeric nMedicalHistory = 1;
54      numeric nFamilyHistory = 2;
55      numeric nSocialHistory = 3;
56      numeric nSOAPSubjective = 4;
57      numeric nSOAPObjective = 5;
58      numeric nSOAPAssessment = 6;
59      numeric nSOAPPlan = 7;
60      numeric nSOAPFollowUp = 8;
61
62      #
63      # intervention weights...
64      # NOTE that all of their zero-th elements have a weight of 0 as filler so
65      # that it will be a 1-based array...
66      numeric nDRPWeights[35] = 0, 3, 3, 3, 4, 5, 3, 3, 3, 3, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3,
        3, 3, 3, 3, 3, 3, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3;
67      numeric nRXTypeWeights[3] = 0, 0, 0;    # weight n/a to RX type so default to 0...
68      numeric nPatientRiskWeights[4] = 0, 3, 2, 1;
69      numeric nProblemTypeWeights[3] = 0, 2, 1;
70      numeric nRecommendationWeights[13] = 0, 3, 3, 3, 3, 3, 3, 3, 1, 1, 3, 2, 1;
71      numeric nActivityWeights[9] = 0, 3, 3, 3, 3, 3, 2, 1, 4, 1;
72      numeric nResultWeights[5] = 0, 3, 2, 1, 1;
73      numeric nTimeWeights[5] = 0, 0, 0, 0, 0;    # n/a
74      numeric nLabValueWeights[5] = 0, 0, 0, 0, 0;
75
76      # list box selections
77      numeric nTotalSelections = 10;
78      #numeric nNONS1 = 0;
79      #numeric nNONS2 = 1;
80      numeric nDRPSelection = 2;
81      numeric nRXTypeSelection = 3;
82      numeric nPatientRiskSelection = 4;
83      numeric nProblemTypeSelection = 5;
84      numeric nRecommendationSelection = 6;
85      numeric nActivitySelection = 7;
86      numeric nResultSelection = 8;
87      numeric nTimeSelection = 9;
88      numeric nDBInterventionSelections[nTotalSelections] = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0;
89
90      #
91      numeric nTotalSearchableRecords;
92      numeric nCurrentDisplayedRecord;
93
94      #
95      numeric nCurrentFrame;
96
97      # preferences
98      numeric fSpeedEntryEnabled;
99      string sWirelessURL;
100     string sLocationID;
101     string sRotationID;
102     string sUserID;
103 end;
104
105
```

```
106  #
107  # PIDS_lists.cpk
108  #
109  ##############################################################################
110  #
111  # PIDS (tm)
112  # Pharmacy Intervention Documentation System
113  # Copyright © 2002  Integrated Software Systems, L.L.C.
114  #
115  ##############################################################################
116  #
117  # Version:  1.0
118  # Date:     1/6/2002
119  # Author:   Todd Viegut & Brad Tice
120  #
121  #
122  # General purpose string arrays for PIDS (tm) pop-up lists.
123  #
124  variables;
125      numeric nAllergiesListSize = 23;
126      string allergies_list[nAllergiesListSize] = # text_entry_allergies_selector
127          "Other",
128          "Acetaminophen",
129          "Aspirin",
130          "Barbiturates",
131          "Benzodiazepines",
132          "Cephalosporins",
133          "Codeine",
134          "Digitalis",
135          "Erythromycin",
136          "Fish Product Derivatives",
137          "Iodine Derivatives",
138          "Local Anesthetics (Amides)",
139          "Local Anesthetics (Esters)",
140          "Nitrofurantoin",
141          "Non-Steroidal Anti-inflammatory Agents (Motrin)",
142          "Penicillins",
143          "Phenothiazines",
144          "Red Dye",
145          "Sulfonamide Derivatives (Sulfa)",
146          "Tartrazine (FD&C Yellow 5)",
147          "Tetracycline",
148          "Vaccines",
149          "No Known Allergies";
150      numeric nDrugsListSize = 207;
151      string drugs_list[nDrugsListSize] = # text_entry_med_history_selector
152          "Accolate",
153          "Accupril",
154          "Accutane",
155          "Aggalat",
156          "Adderall",
157          "Alesse",
158          "Allegra",
159          "Alphagan",
160          "Altace",
161          "Amaryl",
162          "Ambien",
163          "Amoxil",
164          "Aricept",
165          "Arthrotec",
166          "Atrovent",
167          "Augmentin",
168          "Avapro",
169          "Axid",
170          "Azmacort",
171          "Bactroban",
172          "Beconase AQ",
173          "Benzamycin",
174          "Biaxin",
175          "Biaxin Susp",
176          "Buspar",
```

```
177         "Buspar Dividose",
178         "Calan",
179         "Calan SR",
180         "Cardizem",
181         "Cardura",
182         "Ceftin",
183         "Cefzil",
184         "Celebra",
185         "Celecrex",
186         "Cipro",
187         "Claritin",
188         "Claritin RediTabs",
189         "Cleocin-T",
190         "Clinara",
191         "Combivent",
192         "Coumadin",
193         "Covera-HS",
194         "Cozaar",
195         "Darvocet N-100",
196         "Daypro",
197         "Deltasone",
198         "Demadex",
199         "Demulen",
200         "Depakote",
201         "Desogen",
202         "Differin",
203         "Diflucan",
204         "Dilantin",
205         "Diovan",
206         "Dyazide",
207         "Effexor",
208         "Effexor",
209         "Effexor XR",
210         "Elocon",
211         "Estrace",
212         "Estraderm",
213         "Estratest",
214         "Estratest HS",
215         "Evista",
216         "Flonase",
217         "Flovent",
218         "Floxin",
219         "Fosamax",
220         "Glucophage",
221         "Glucotrol",
222         "Glynase",
223         "Humulin 70/30",
224         "Humulin N",
225         "Humulin R",
226         "Hytrin",
227         "Hyzaar",
228         "Imdur",
229         "Imitrex",
230         "K-Dur",
231         "K-Dur",
232         "K-Lonopin",
233         "Lac-Hydrin",
234         "Lamisil",
235         "Lamisil Oral",
236         "Lanoxin",
237         "Lasix",
238         "Lescol",
239         "Levaquin",
240         "Lipitor",
241         "Lo/Ovral",
242         "Lodine",
243         "LodineXL",
244         "Loestrin",
245         "Lopressor",
246         "Lorabid",
247         "Lotensin",
```

```
248         "Lotrel",
249         "Lotriscne",
250         "Macrobid",
251         "MetroGel Vaginal",
252         "Mevacor Risperdal",
253         "Miacalcin",
254         "Monopril",
255         "Nasacort",
256         "Nasacort AQ",
257         "Nasonex",
258         "Neurontin",
259         "Nitro-Dur",
260         "Nitrostat",
261         "Nizoral Shampoo",
262         "Nizoral tablets",
263         "Nizoral Topical",
264         "None",
265         "Nordette",
266         "Norvasc",
267         "Novolin 70/30",
268         "Novolin N",
269         "Novolin R",
270         "One Touch Test Strips",
271         "Ortho-Cept",
272         "Ortho-Cyclen",
273         "Ortho-Novum",
274         "Ortho Tri-Cyclen",
275         "Other",
276         "Oxycontin",
277         "Paxil",
278         "Pepcid",
279         "Phenergan",
280         "Plendil",
281         "Pravachol",
282         "Prednisone",
283         "Premarin",
284         "Premarin Vaginal",
285         "Premphase",
286         "Prempro",
287         "Prenate Ultra",
288         "Prevacid",
289         "Prilosec",
290         "Prinivil",
291         "Prinzide",
292         "Procardia",
293         "Propulsid",
294         "Proscar",
295         "Proventil",
296         "Proventil HFA",
297         "Provera",
298         "Prozac",
299         "Relafen",
300         "Remeron",
301         "Retin-A",
302         "Rezulin",
303         "Rhinocort",
304         "Ritalin",
305         "Serevent",
306         "Serzone",
307         "Singulair",
308         "Skelaxin",
309         "Suprax",
310         "Synthroid",
311         "Tapazole",
312         "Tegretol",
313         "Tenormin",
314         "Terazol 3",
315         "Terazol 7",
316         "Tiazac",
317         "Ticlid",
318         "Timoptic",
```

```
319         "Timoptic XE",
320         "Tobradex",
321         "Toprol XL",
322         "Tri-Levlen",
323         "Triphasil",
324         "Trusopt",
325         "Tussionex",
326         "Ultram",
327         "Valium",
328         "Valtrex",
329         "Vancenase",
330         "Vanceril",
331         "Vaseretic",
332         "Vasotec",
333         "Ventolin",
334         "Verelan",
335         "Viagra",
336         "Vicodin",
337         "Vicodin ES",
338         "Vicoprofen",
339         "Vioxx",
340         "Voltaren XR",
341         "Wellbutrin",
342         "Wellbutrin",
343         "Westcort",
344         "Xalatan",
345         "Xanax",
346         "Zantac",
347         "Zaroxolyn",
348         "Zestoretic",
349         "Zestril",
350         "Ziac",
351         "Zithromax",
352         "Zocor",
353         "Zoloft",
354         "Zovirax Topical",
355         "Zyban",
356         "Zyprexa",
357         "Zyprexa",
358         "Zyrtec";
359     numeric nDiseasesListSize = 41;
360     string diseases_list[nDiseasesListSize] = # text_entry_family_history_selector
361         "None",
362         "Unknown",
363         "Acne or Psoriasis",
364         "AIDS",
365         "Allergies (Drug Related)",
366         "Allergies (Non-Drug Related)",
367         "Alzheimer's Disease",
368         "Angina",
369         "Anxiety Disorders",
370         "Asthma",
371         "Bipolar Disorders",
372         "Cancer",
373         "Congestive Heart Failure",
374         "Constipation & Diarrhea",
375         "Diabetes Mellitus",
376         "Epilepsy",
377         "GERD",
378         "Glaucoma",
379         "Gout",
380         "Headache Disorders",
381         "Heart Arrythmias",
382         "Hormone Replacement Therapy",
383         "Hyperlipidemia",
384         "Hypertension",
385         "Infectious Disease",
386         "Inflammatory Bowel Disease",
387         "Multiple Sclerosis",
388         "Nausea & Vomiting",
389         "Nutrition Related Disorders",
```

```
390         "Osteoarthritis",
391         "Osteoporosis",
392         "Other",
393         "Pain Management",
394         "Pancreatitis",
395         "Peptic Ulcer Disease",
396         "Premenstrual Syndrome",
397         "Rheumatoid Arthritis",
398         "Sleep Disorders",
399         "Stroke",
400         "Substance-Related Disorders",
401         "Thyroid Disorders";
402
403     string social_list[5] = # text_entry_social_history_selector
404         "Caffeine",
405         "Diet and Exercise",
406         "EtOH",
407         "Tobacco",
408         "Illicit Drug Use";
409
410     #
411     # Lab Values lists...
412     #
413     string blood_pressure_list[3] =
414         "[Please select one]",
415         "Systolic Blood Pressure",
416         "Diastolic Blood Pressure";
417     string respiratory_value_list[4] =
418         "[Please select one]",
419         "Respiratory Rate (per minute)",
420         "PEFR (% personal best)",
421         "PEFR (% predicted)";
422     string blood_sugar_values[4] =
423         "[Please select one]",
424         "Fasting glucose (mg/dL)",
425         "Non-Fasting glucose (mg/dl)",
426         "Hemoglobin A1c (%)";
427     string lipid_values_list[6] =
428         "[Please select one]",
429         "Total Cholesterol",
430         "HDL",
431         "LDL",
432         "Triglycerides",
433         "TC/HDL ratio";
434     string osteoporosis_values_list[3] =
435         "[Please select one]",
436         "T-Score",
437         "Z-Score";
438     string liver_function_tests_list[4] =
439         "[Please select one]",
440         "ALT",
441         "AST",
442         "Alk Phos";
443     string electrolytes_list[5] =
444         "[Please select one]",
445         "Na+",
446         "K+",
447         "Cl-",
448         "PO4";
449     string anticoagulation_values_list[2] =
450         "[Please select one]",
451         "INR";
452     string renal_function_list[4] =
453         "[Please select one]",
454         "BUN",
455         "Cr",
456         "Creatinine Clearance (ClCr)";
457     string hematology_list[4] =
458         "[Please select one]",
459         "White Blood Cells",
460         "Segs",
```

```
461          "Bands";
462      #
463      # Lab values list...
464      string completed_lab_values[20] = "";
465  end;
466
467
```

```
#
# PIDS.csl
#
##############################################################################
#
# PIDS (tm)
# Pharmacy Intervention Documentation System
# Copyright © 2002  Integrated Software Systems, L.L.C.
#
##############################################################################
#
# Version:  1.0
# Date:     8/15/2001
# Author:   Todd Viegut & Brad Tice
#
#
# MAIN SOURCE CODE FILE FOR PIDS (tm)
#

#
include "PIDS_date_lib.cpk";              # include intervention date routines
include "casler26.cpk";                  # include the CASL error definitions
include "casl_net_errors.cpk";           # include CASL network errors
include "PIDS_globals.cpk";              # GLOBAL variables
include "PIDS_lists.cpk";                # include the string lists for PIDS selectors
include "PIDS_sync.cpk";                 # synchronization functions include file
include "PIDS_misc.cpk";                 # miscellaneous function for PIDS
include "PIDS_wizards.cpk";
include "XML_parser.cpk";

#
#-----------------------------------------------
# Standard functions for moving data between
# interventions and DB records and vice versa.
# Also included is the RESET method to clear
# all fields in order to apply another
# intervention.
#-----------------------------------------------
#
function ResetFieldsForNextIntervention;
    variables;
        numeric nIndex;
    end;
    #
    fUpdateMode = false;
    #
    nLabValuesCreated = 0;
    nDRPSelected = 0;
    nDBSynced = 0;
    #
    # Clear start fields
    txLocationID.display = sLocationID;
    #hide intervention_date_button;
    intervention_date_button.display = GetCurrentInterventionDate;
    txRotationID.display = sRotationID;
    txUserID.display = sUserID;
    txPrescribedDrug.display = "";
    txDisease.display = "";
    txPrescriber.display = "";
    #
    # Clear db numeric fields...
    for nIndex = 0, nIndex < nTotalSelections;
        nDBInterventionSelections[nIndex] = 0;
    next nIndex + 1;
    #
    # Clear selectors...
    for nIndex = 0, nIndex < nTotalFrames;
        hide objInterventionSelectors[nIndex];
        objInterventionSelectors[nIndex].selected = 0;
    next nIndex + 1;
    #
```

```
539     # Clear DRP selectors...
540     for nIndex = 0, nIndex < nTotalDRPFrames;
541         hide objDRPSelectors[nIndex];
542         objDRPSelectors[nIndex].selected = 0;
543     next nIndex + 1;
544     #
545     txPatientID.display = "";
546     #
547     # Clear db memo fields
548     for nIndex = 0, nIndex < nTotalTextEntries;
549         sDBTextEntryMemos[nIndex] = "";
550     next nIndex + 1;
551     #
552     # Clear text fields... etc.
553     for nIndex = 0, nIndex < nTotalTextEntries;
554         objTextEntryTextFields[nIndex].display = "";
555     next nIndex + 1;
556     #
557     # Clear lab values list...
558     call ClearLabValuesList;
559     #
560     # Miscellaneous fields...
561     txtLabValues.display = "";
562 end;
563
564 function MoveInterventionToDBFields;
565     variables;
566         string sValue;
567         numeric nIndex;
568     end;
569
570     # set primary key index...
571     nDBPIDSKeyField = GetInterventionKeyId;
572     nDBSynced = 0;
573     #
574     sDBLocationID = txLocationID.display;
575     sDBInterventionDate = intervention_date_button.display;
576     sDBRotationID = txRotationID.display;
577     sDBUserID = txUserID.display;
578     sDBPrescribedDrug = txPrescribedDrug.display;
579     sDBDisease = txDisease.display;
580     sDBPrescriber = txPrescriber.display;
581     #
582     # set db numeric fields...
583     nDBDrugRelatedProblemField = nDBInterventionSelections[nDRPSelection];
584     nDBRXTypeField = nDBInterventionSelections[nRXTypeSelection];
585     nDBPatientRiskField = nDBInterventionSelections[nPatientRiskSelection];
586     nDBProblemTypeField = nDBInterventionSelections[nProblemTypeSelection];
587     nDBRecommendationField = nDBInterventionSelections[nRecommendationSelection];
588     nDBActivityField = nDBInterventionSelections[nActivitySelection];
589     nDBResultField = nDBInterventionSelections[nResultSelection];
590     nDBTimeField = nDBInterventionSelections[nTimeSelection];
591     #
592     #
593     for nIndex = 0, nIndex < nTotalTextEntries;
594         get objTextEntryTextFields[nIndex], sValue;
595         sDBTextEntryMemos[nIndex] = sValue;
596     next nIndex + 1;
597     #
598     # set db memo fields
599     sDBPatientIDField = txPatientId.display;
600     sDBAllergiesField = sDBTextEntryMemos[nAllergies];
601     sDBMedicalHistoryField = sDBTextEntryMemos[nMedicalHistory];
602     sDBFamilyHistoryField = sDBTextEntryMemos[nFamilyHistory];
603     sDBSocialHistoryField = sDBTextEntryMemos[nSocialHistory];
604     #
605     sDBSOAPSubjectiveField = sDBTextEntryMemos[nSOAPSubjective];
606     sDBSOAPObjectiveField = sDBTextEntryMemos[nSOAPObjective];
607     sDBSOAPAssessmentField = sDBTextEntryMemos[nSOAPAssessment];
608     sDBSOAPPlanField = sDBTextEntryMemos[nSOAPPlan];
609     sDBSOAPFollowUpField = sDBTextEntryMemos[nSOAPFollowUp];
```

```
610     #
611     sDBLabValues = txtLabValues.display;
612     nDBDOIScore = CalculateDOIScore;
613 end;
614
615 function MoveDBFieldsToIntervention;
616     variables;
617         numeric nIndex;
618     end;
619     #
620     # set primary key index...
621     nInterventionKeyId = nDBPIDSKeyField;
622     #
623     txLocationID.display = sDBLocationID;
624     hide intervention_date_button;
625     intervention_date_button.display = sDBInterventionDate;
626     txRotationID.display = sDBRotationID;
627     txUserID.display = sDBUserID;
628     txPrescribedDrug.display = sDBPrescribedDrug;
629     txDisease.display = sDBDisease;
630     txPrescriber.display = sDBPrescriber;
631     #
632     nDRPSelected = GetDRPTypeIndex(nDBDrugRelatedProblemField);
633     #
634     nIndex = 0;
635     #
636     for nIndex = 0, nIndex < nTotalDRPFrames;
637         if (nIndex = nDRPSelected);
638             objDRPSelectors[nIndex - 1].selected =
                GetDRPTypeSubIndex(nDBDrugRelatedProblemField);
639         else;
640             objDRPSelectors[nIndex].selected = 0;
641         end_if;
642     next nIndex + 1;
643     #
644     # Set DB numeric fields...
645     nDBInterventionSelections[nDRPSelection] = nDBDrugRelatedProblemField;
646     nDBInterventionSelections[nRXTypeSelection] = nDBRXTypeField;
647     nDBInterventionSelections[nPatientRiskSelection] = nDBPatientRiskField;
648     nDBInterventionSelections[nProblemTypeSelection] = nDBProblemTypeField;
649     nDBInterventionSelections[nRecommendationSelection] = nDBRecommendationField;
650     nDBInterventionSelections[nActivitySelection] = nDBActivityField;
651     nDBInterventionSelections[nResultSelection] = nDBResultField;
652     nDBInterventionSelections[nTimeSelection] = nDBTimeField;
653     #
654     objInterventionSelectors[nDRPSelection].selected = nDRPSelected;
655     objInterventionSelectors[nRXTypeSelection].selected =
        nDBInterventionSelections[nRXTypeSelection];
656     objInterventionSelectors[nPatientRiskSelection].selected =
        nDBInterventionSelections[nPatientRiskSelection];
657     objInterventionSelectors[nProblemTypeSelection].selected =
        nDBInterventionSelections[nProblemTypeSelection];
658     objInterventionSelectors[nRecommendationSelection].selected =
        nDBInterventionSelections[nRecommendationSelection];
659     objInterventionSelectors[nActivitySelection].selected =
        nDBInterventionSelections[nActivitySelection];
660     objInterventionSelectors[nResultSelection].selected =
        nDBInterventionSelections[nResultSelection];
661     objInterventionSelectors[nTimeSelection].selected =
        nDBInterventionSelections[nTimeSelection];
662     #
663     # set db memo fields
664     txPatientId.display = sDBPatientIDField;
665     sDBTextEntryMemos[nAllergies] = sDBAllergiesField;
666     sDBTextEntryMemos[nMedicalHistory] = sDBMedicalHistoryField;
667     sDBTextEntryMemos[nFamilyHistory] = sDBFamilyHistoryField;
668     sDBTextEntryMemos[nSocialHistory] = sDBSocialHistoryField;
669     #
670     sDBTextEntryMemos[nSOAPSubjective] = sDBSOAPSubjectiveField;
671     sDBTextEntryMemos[nSOAPObjective] = sDBSOAPObjectiveField;
672     sDBTextEntryMemos[nSOAPAssessment] = sDBSOAPAssessmentField;
```

```
673         sDBTextEntryMemos[nSOAPPlan] = sDBSOAPPlanField;
674         sDBTextEntryMemos[nSOAPFollowUp] = sDBSOAPFollowUpField;
675         #
676         nIndex = 0;
677         #
678         for nIndex = 0, nIndex < nTotalTextEntries;
679             objTextEntryTextFields[nIndex].display = sDBTextEntryMemos[nIndex];
680         next nIndex + 1;
681         #
682         txtLabValues.display = sDBLabValues;
683     end;
684
685     #
686     #-------------------------------------------------
687     # MENU OBJECT DEFINITIONS
688     #-------------------------------------------------
689     #
690     menu_top mtOptions;
691         display "Options";
692     end;
693
694     menu_item miSearch, mtOptions;
695         display "Search for intervention...";
696     end;
697
698     menu_item miSync, mtOptions;
699         display "Perform wireless sync...";
700     end;
701
702     #menu_item miPurgeDB, mtOptions;
703     #   display "Purge interventions...";
704     #end;
705
706     menu_top mtSetup;
707         display "Setup";
708     end;
709
710     menu_item miPreferences, mtSetup;
711         display "Set preferences...";
712     end;
713
714     #menu_item miDeletePIDSDatabase, mtSetup;
715     #   display"Delete database...";
716     #end;
717
718     function SelectInterventionDate;
719         sDBInterventionDate = GetCurrentInterventionDate;
720         intervention_date_button.display = sDBInterventionDate;
721     end;
722
723     function DrugRelatedProblemTypeSelected;
724         variables;
725             numeric nItemIndex;
726         end;
727         get drug_related_problem_selector, nItemIndex;
728         nDRPSelected = nItemIndex;
729         if (nItemIndex > 0) and (nItemIndex <= nTotalDRPFrames);
730             hide drug_related_problem_frame;
731             show objDRPSelectors[nItemIndex - 1];    # - 1 because of 0 based arrays...
732             show objDRPFrames[nItemIndex - 1];  # - 1 because of 0 based arrays...
733         end_if;
734     end;
735
736     function DRPPreviousForm;
737         hide objDRPFrames[nDRPSelected - 1];     # - 1 because of 0 based arrays...
738         hide objDRPSelectors[nDRPSelected - 1]; # - 1 because of 0 based arrays...
739         show drug_related_problem_frame;
740     end;
741
742     function DRPNextForm;
743         variables;
```

```
744         object objSelector;
745         numeric nDRPSelectedItemIndex;
746         #numeric nSelectedItemIndex;
747     end;
748     #
749     # Set the intervention selector DB values...
750     objSelector = objDRPSelectors[nDRPSelected - 1];
751     nDRPSelectedItemIndex = objSelector.selected;
752     if (nDRPSelectedItemIndex > 0);
753         nDBInterventionSelections[nCurrentFrame] = nDRPSelectionListOffsets[nDRPSelected]
        + nDRPSelectedItemIndex;
754         #
755         # Drop the DRP chosen into the Assessment field of the SOAP screen...
756         if (objPIDSFrames[nCurrentFrame] = patient_info_frame) and
            (length(txAssessment.display) = 0);
757             txAssessment.display =
                GetDrugRelatedProblemData(nDBInterventionSelections[nCurrentFrame]);
758         end_if;
759         #
760         # Now switch frames...
761         hide objDRPFrames[nDRPSelected - 1];    # - 1 because of 0 based arrays...
762         hide objDRPSelectors[nDRPSelected - 1]; # - 1 because of 0 based arrays...
763         nCurrentFrame = nCurrentFrame + 1;
764         call ShowFrame;
765     end_if;
766 end;
767
768 function PreviousForm;
769     if (nCurrentFrame > 0);
770         if (nCurrentFrame > 1);
771             call HideFrame;
772             nCurrentFrame = nCurrentFrame - 1;
773         end_if;
774         call ShowFrame;
775     end_if;
776 end;
777
778 function NextForm;
779     variables;
780         object objSelector;
781         numeric nSelectedItemIndex;
782     end;
783     if (nCurrentFrame < (nTotalFrames - 1));
784         if (nContainsSelector[nCurrentFrame]);
785             objSelector = objInterventionSelectors[nCurrentFrame];
786             nSelectedItemIndex = objSelector.selected;
787             if (nSelectedItemIndex > 0);
788                 if (nCurrentFrame = DRP_FRAME_NUMBER);
789                     call DrugRelatedProblemTypeSelected;
790                 else;
791                     #
792                     # Set the intervention selector DB values...
793                     nDBInterventionSelections[nCurrentFrame] = nSelectedItemIndex;
794                     #
795                     # Now switch frames...
796                     call HideFrame;
797                     nCurrentFrame = nCurrentFrame + 1;
798                     call ShowFrame;
799                 end_if;
800             end_if;
801         else;
802             if (objPIDSFrames[nCurrentFrame] = patient_info_frame) and
                (length(txSubjective.display) = 0);
803                 txSubjective.display = GetPatientInfoData();
804             end_if;
805             #
806             # UNCOMMENT LATER!!!!
807             #
808             if (objPIDSFrames[nCurrentFrame] = patient_info_frame) and
                (length(txAssessment.display) = 0);
809                 txAssessment.display =
```

```
                    GetDrugRelatedProblemData(nDBInterventionSelections[DRP_FRAME_NUMBER]);
            end_if;
            #
            # Drop the DRP chosen into the Assessment field of the SOAP screen...
            #
            call HideFrame;
            nCurrentFrame = nCurrentFrame + 1;
            call ShowFrame;
        end_if;
    end_if;
end;

#
#-------------------------------------------------
# INVOKER functions for the seven frames that the
# drug related problem frame directs to...
#-------------------------------------------------
#

function ItemSelected;
    variables;
        object objSelector;
        numeric nDRPSelectedItemIndex;
        numeric nSelectedItemIndex;
    end;
    if fSpeedEntryEnabled;
        if (nCurrentFrame < (nTotalFrames - 1));
            objSelector = objInterventionSelectors[nCurrentFrame];
            nSelectedItemIndex = objSelector.selected;
            if (nSelectedItemIndex > 0);
                if (nCurrentFrame = DRP_FRAME_NUMBER) and (invoker.object_string = "DRP");
                    call DrugRelatedProblemTypeSelected;
                else;
                    #
                    # Set the intervention selector DB values...
                    if nCurrentFrame = DRP_FRAME_NUMBER;
                        objSelector = objDRPSelectors[nDRPSelected - 1];
                        nDRPSelectedItemIndex = objSelector.selected;
                        nDBInterventionSelections[nCurrentFrame] =
                        nDRPSelectionListOffsets[nDRPSelected] + nDRPSelectedItemIndex;
                    else;
                        nDBInterventionSelections[nCurrentFrame] = nSelectedItemIndex;
                    end_if;
                    #
                    # Now switch frames...
                    call HideFrame;
                    nCurrentFrame = nCurrentFrame + 1;
                    call ShowFrame;
                end_if;
            end_if;
        end_if;
    end_if;
end;

#
# Search Screen Function Definitions:
#
function UpdateSearchDisplay(numeric nRecord);
    if ((nTotalSearchableRecords = 0) or ((nRecord = 1) and (nTotalSearchableRecords =
    1)));
        previous_record_button.display = char(5);
        next_record_button.display = char(6);
    else_if (nRecord = 1);
        previous_record_button.display = char(5);
        next_record_button.display = char(3);
    else_if (nRecord = nTotalSearchableRecords);
        previous_record_button.display = char(2);
        next_record_button.display = char(6);
    else;
        previous_record_button.display = char(2);
        next_record_button.display = char(3);
```

```
878     end_if;
879     record_set_label.display = string(nRecord, "#") + " of " +
        string(nTotalSearchableRecords, "#");
880 end;
881
882 function ClearSearchDisplay;
883     variables;
884         numeric nIndex;
885     end;
886     #
887     nCurrentDisplayedRecord = 0;
888     nTotalSearchableRecords = 0;
889     UpdateSearchDisplay(nCurrentDisplayedRecord);
890     #
891     hide search_edit_button;
892     hide search_delete_intervention_button;
893     #
894     nIndex = 0;
895     #
896     # change all button bitmaps to "empty" state...
897     for nIndex = 0, nIndex < nTotalTextEntries;
898         objSearchTextEntryButtons[nIndex].display = char(11);
899     next nIndex + 1;
900     #
901     txSearchPatientID.display = "";
902     txSearchDate.display = "";
903     txSearchDRPDescription.display = "";
904     txSearchRxType.display = "";
905     txSearchPatientRisk.display = "";
906     txSearchProblemType.display = "";
907     txSearchPrescribedDrug.display = "";
908     txSearchRecommendation.display = "";
909     txSearchActivity.display = "";
910     txSearchResult.display = "";
911     txSearchTime.display = "";
912     txSearchDOIScore.display = "";
913 end;
914
915 function IsRecordSynced(numeric nRecord) as numeric;
916     if (dba_ffOpenDB(dbPidsXML, "dbPidsXML"));
917         if (dba_ffReadRecordFromRecNo(dbPidsXML, nRecord));
918             IsRecordSynced = nDBXMLSynced;
919         else;
920             IsRecordSynced = -1; # Indicated an error occurred...
921         end_if;
922         dba_CloseDB(dbPidsXML);
923     else;
924         IsRecordSynced = -1; # Indicated an error occurred...
925     end_if;
926 end;
927
928 function PopulateSearchScreen(numeric nRecord);
929     variables;
930         object oText;
931         string sCurrentChar;
932         string sValue;
933         numeric nIndex;
934         numeric nDRPSubIndex;
935         numeric nSynced;
936     end;
937     #
938     nSynced = IsRecordSynced(nRecord);
939     if (dba_ffOpenDB(dbPids, "dbPids"));
940         if (dba_ffReadRecordFromRecNo(dbPids, nRecord));
941             #
942             if (not (nSynced = 0));
943                 hide search_edit_button;
944             else;
945                 show search_edit_button;
946             end_if;
947             show search_delete_intervention_button;
```

```
948             #
949             nIndex = GetDRPTypeIndex(nDBDrugRelatedProblemField);
950             #
951             nDRPSubIndex = GetDRPTypeSubIndex(nDBDrugRelatedProblemField);
952             #
953             if nIndex = 1;
954                 txSearchDRPDescription.display =
                    unnecessary_drug_therapy_list[nDRPSubIndex];
955             else_if nIndex = 2;
956                 txSearchDRPDescription.display = wrong_drug_list[nDRPSubIndex];
957             else_if nIndex = 3;
958                 txSearchDRPDescription.display = dosage_too_low_list[nDRPSubIndex];
959             else_if nIndex = 4;
960                 txSearchDRPDescription.display = adverse_drug_reaction_list[nDRPSubIndex];
961             else_if nIndex = 5;
962                 txSearchDRPDescription.display = dosage_too_high_list[nDRPSubIndex];
963             else_if nIndex = 6;
964                 txSearchDRPDescription.display =
                    inappropriate_compliance_list[nDRPSubIndex];
965             else_if nIndex = 7;
966                 txSearchDRPDescription.display = need_therapy_list[nDRPSubIndex];
967             end_if;
968             #
969             txSearchPatientID.display = sDBPatientIDField;
970             #
971             txSearchDate.display = sDBInterventionDate;
972             #
973             txSearchRxType.display = rx_type_list[nDBRXTypeField];
974             #
975             txSearchPatientRisk.display = patient_risk_type_list[nDBPatientRiskField];
976             #
977             txSearchProblemType.display = problem_type_list[nDBProblemTypeField];
978             #
979             txSearchPrescribedDrug.display = sDBPrescribedDrug;
980             #
981             txSearchRecommendation.display =
                intervention_recommendation_list[nDBRecommendationField];
982             #
983             txSearchActivity.display = intervention_activity_list[nDBActivityField];
984             #
985             txSearchResult.display = intervention_result_list[nDBResultField];
986             #
987             txSearchTime.display = intervention_time_list[nDBTimeField];
988             #
989             txSearchDOIScore.display = string(nDBDOIScore, "#");
990             #
991             search_allergies_button.object_string = sDBAllergiesField;
992             search_med_history_button.object_string = sDBMedicalHistoryField;
993             search_family_history_button.object_string = sDBFamilyHistoryField;
994             search_social_history_button.object_string = sDBSocialHistoryField;
995             #
996             search_subjective_button.object_string = sDBSOAPSubjectiveField;
997             search_objective_button.object_string = sDBSOAPObjectiveField;
998             search_assessment_button.object_string = sDBSOAPAssessmentField;
999             search_plan_button.object_string = sDBSOAPPlanField;
1000            search_follow_up_button.object_string = sDBSOAPFollowUpField;
1001
1002            #
1003            nIndex = 0;
1004            #
1005            for nIndex = 0, nIndex < nTotalTextEntries;
1006                oText = objSearchTextEntryButtons[nIndex];
1007                sValue = oText.object_string;
1008                if (length(sValue) > 0);
1009                    get objSearchTextEntryButtons[nIndex], sCurrentChar;
1010                    if (sCurrentChar <> char(9));
1011                        put objSearchTextEntryButtons[nIndex], char(9);
1012                    end_if;
1013                else;
1014                    get objSearchTextEntryButtons[nIndex], sCurrentChar;
1015                    if (sCurrentChar <> char(11));
```

```
                        put objSearchTextEntryButtons[nIndex], char(11);
                    end_if;
                end_if;
                next nIndex + 1;
                # = nDBDOIScore;
            else;
                call ClearSearchDisplay;
                ErrorMsg("Unable to read intervention from the PIDS database.");
            end_if;
            dba_CloseDB(dbPids);
        else;
            call ClearSearchDisplay;
            ErrorMsg("Unable to open PIDS intervention database.");
        end_if;
    end;

    function InitSearch();
        variables;
            numeric nIndex;
        end;
        #
        call ClearSearchDisplay;
        #
        if (dba_ffOpenDB(dbPids, "dbPids"));
            #
            # Retrieve the total records currently in the database...
            nIndex = 0;
            seek_start dbPids, 0;
            get_fields dbPids;
            #
            while (errorcode = ce_no_error);
                nIndex = nIndex + 1;
                get_fields dbPids;
            end_while;
            #
            if (errorcode <> ce_eof);
                ErrorMsg("Problem reading interventions from the PIDS database.");
            else;
                nTotalSearchableRecords = nIndex;
                if (nTotalSearchableRecords > 0);
                    nCurrentDisplayedRecord = 1;
                    call UpdateSearchDisplay(nCurrentDisplayedRecord);
                    call PopulateSearchScreen(nCurrentDisplayedRecord - 1);
                end_if;
            end_if;
            dba_CloseDB(dbPids);
        else;
            ErrorMsg("Unable to open PIDS intervention database.");
        end_if;
    end;
    #
    #-------------------------------------------------
    # Previous and next record functions for the
    # search screen...
    #-------------------------------------------------
    #
    function PreviousRecord;
        if (nCurrentDisplayedRecord > 1);
            nCurrentDisplayedRecord = nCurrentDisplayedRecord - 1;
            call UpdateSearchDisplay(nCurrentDisplayedRecord);
            call PopulateSearchScreen(nCurrentDisplayedRecord - 1);
        end_if;
    end;

    function NextRecord;
        if (nCurrentDisplayedRecord < nTotalSearchableRecords);
            nCurrentDisplayedRecord = nCurrentDisplayedRecord + 1;
            call UpdateSearchDisplay(nCurrentDisplayedRecord);
            call PopulateSearchScreen(nCurrentDisplayedRecord - 1);
        end_if;
    end;
```

```
1087
1088  function ShowTextEntryScreen;
1089      variables;
1090          string sValue;
1091          numeric nIndex;
1092          numeric nTextEntryIndex;
1093      end;
1094      #
1095      # string sTitles[nTotalTextEntries] = "Allergies", "Med History", "Family History",
          "Social History",
1096      #                                    "Subjective", "Objective", "Assessment", "Plan",
          "Follow Up";
1097      # Only these 4 text entry screens have pop up capability...
1098      #
1099      for nIndex = 0, nIndex < nTotalTextEntries;
1100          if objTextEntryButtons[nIndex] = invoker;
1101              nTextEntryIndex = nIndex;
1102              #
1103              text_entry_frame.display = sTitles[nIndex];
1104              accept_text_entry_button.object_string = sTitles[nIndex];
1105              #
1106              generic_selector_button.object_string = sTitles[nIndex];
1107              generic_selector_label.display = sTitles[nIndex];
1108              #
1109              get objTextEntryTextFields[nIndex], sValue;
1110              txEntry.display = sValue;
1111          end_if;
1112      next nIndex + 1;
1113      #
1114      call HideFrame;
1115      show text_entry_frame;
1116      if ((nTextEntryIndex = 0) or
1117          (nTextEntryIndex = 1) or
1118          (nTextEntryIndex = 2) or
1119          (nTextEntryIndex = 3));
1120          show generic_selector_label;
1121          show generic_selector_button;
1122      else;
1123          hide generic_selector_label;
1124          hide generic_selector_button;
1125      end_if;
1126  end;
1127
1128  function AcceptTextEntry;
1129      variables;
1130          numeric nIndex;
1131      end;
1132      for nIndex = 0, nIndex < nTotalTextEntries;
1133          if sTitles[nIndex] = accept_text_entry_button.object_string;
1134              objTextEntryTextFields[nIndex].display = txEntry.display;
1135          end_if;
1136      next nIndex + 1;
1137      hide text_entry_frame;
1138      call ShowFrame;
1139  end;
1140
1141  function CancelTextEntry;
1142      hide text_entry_frame;
1143      call ShowFrame;
1144  end;
1145
1146  function miSync;
1147      variables;
1148          numeric selection;
1149      end;
1150      selection = message_box(1,"Perform Wireless Sync", "Would you like to perform a
          wireless sync via the web?", "Yes","Cancel","");
1151      if (selection = 0);
1152          call SyncAllInterventions(sWirelessURL);
1153      end_if;
1154  end;
```

```
1155
1156  function miPreferences;
1157      call HideFrame;
1158      hide intervention_search_frame;
1159      put txPrefsWirelessSyncURL, sWirelessURL;
1160      put txPrefsLocationID, sLocationID;
1161      put txPrefsRotationID, sRotationID;
1162      put txPrefsUserID, sUserID;
1163      put cbPrefsSpeedEntryCheckbox, fSpeedEntryEnabled;
1164      show pids_preferences_frame;
1165  end;
1166
1167  function miDeletePIDSDatabase;
1168      if (message_box(1,"Delete PIDS Database", "Are you sure you want to delete PIDS (al_
          data will be lost)?", "Yes","Cancel","") = 0);
1169          dba_DeleteDB(dbPids, "dbPids");
1170
1171      end_if;
1172      if (message_box(1,"Delete Preferences Database", "Are you sure you want to delete
          preferences (all data will be lost)?", "Yes","Cancel","") = 0);
1173          dba_DeleteDB(dbPidsPrefs, "dbPidsPrefs");
1174      end_if;
1175  end;
1176
1177  function miSearch;
1178      call HideFrame();
1179      hide pids_preferences_frame;
1180      show intervention_search_frame;
1181      call InitSearch();
1182  end;
1183
1184  #
1185  #-------------------------------------------------
1186  # Auxiliary invoker methods...
1187  #-------------------------------------------------
1188  #
1189  function AcceptPreferences;
1190      variables;
1191          numeric nReturnCode;
1192      end;
1193      nReturnCode = SavePreferences;
1194      #
1195      hide pids_preferences_frame;
1196      call ShowFrame;
1197  end;
1198
1199  function SaveAndEnterAnother;
1200      variables;
1201          numeric nKey;
1202          numeric nReturnCode;
1203      end;
1204      nKey = GetInterventionKeyId;
1205      call MoveInterventionToDBFields;
1206      nReturnCode = SaveInterventionRecord(nKey, sWirelessURL);
1207      call HideFrame;
1208      call ResetFieldsForNextIntervention;
1209      nCurrentFrame = 1;
1210      call ShowFrame;
1211      show intervention_date_button;
1212  end;
1213
1214  function DeleteChosenIntervention;
1215      variables;
1216          numeric fDeleted;
1217      end;
1218      if (message_box(1,"Delete Intervention", "Are you sure you want to delete the selected
          intervention?", "Yes","Cancel","") = 0);
1219          # delete the record corresponding to the key .
1220          fDeleted = DeleteInterventionRecord(nDBPIDSKeyField);
1221          #
1222          if (fDeleted);
```

```
            nTotalSearchableRecords = nTotalSearchableRecords - 1;
            if (nTotalSearchableRecords > 0);
                if (nCurrentDisplayedRecord > nTotalSearchableRecords);
                    nCurrentDisplayedRecord = nCurrentDisplayedRecord - 1;
                end_if;
                call UpdateSearchDisplay(nCurrentDisplayedRecord);
                call PopulateSearchScreen(nCurrentDisplayedRecord - 1);
            else;
                call ClearSearchDisplay;
            end_if;
        end_if;
    end_if;
end;

function ContinueEditing;
    call HideFrame;
    nCurrentFrame = 1;
    call ShowFrame;
end;

function CancelPreferences;
    hide pids_preferences_frame;
    call ShowFrame;
end;

#
# ------------------------------------------------
# Functions related to SEARCH text entry screens...
#------------------------------------------------
#
function ShowSearchTextEntryScreen;
    variables;
        numeric nIndex;
    end;
    for nIndex = 0, nIndex < nTotalTextEntries;
        if objSearchTextEntryButtons[nIndex] = invoker;
            search_text_entry_frame.display = sTitles[nIndex];
            txSearchTextEntry.display = invoker.object_string;
        end_if;
    next nIndex + 1;
    hide intervention_search_frame;
    show search_text_entry_frame;
end;

function OkSearchTextEntry;
    hide search_text_entry_frame;
    show intervention_search_frame;
end;

function EditIntervention;
    fUpdateMode = true;
    hide intervention_search_frame;
    nCurrentFrame = 1;
    call ShowFrame;
    call MoveDBFieldsToIntervention;
    show intervention_date_button;
end;

function ExitSearch;
    hide intervention_search_frame;
    call ShowFrame;
end;

#
#-------------------------------------------------------------
#       STARTUP AND SHUTDOWN INVOKER FUNCTIONS
#-------------------------------------------------------------
#
# invoked at program startup
#
```

```
1294 function startup;
1295     call Init;
1296     nCurrentFrame = 0;
1297     call ResetFieldsForNextIntervention;
1298     if (length(sWirelessURL) > 0);
1299         show PIDS_intro_frame;
1300     else;
1301         call miPreferences;
1302     end_if;
1303 end;
1304
1305 #
1306 # invoked at program shutdown,
1307 # closes pids and preferences db
1308 #
1309 function shutdown;
1310     call dba_CloseDB(dbPids);
1311     call dba_CloseDB(dbPidsPrefs);
1312 end;
1313
1314
```

```
#
# CASL_NET_ERRORS.CPK - Package containing CASL / PalmOS error codes/functions (Version
3.2)
#
# This file is a CASL package intended for inclusion in CASL programs
# that will make use of the PalmOS networking error codes and their associated
# strings.
#
# NOTE: The information regarding network error code number to string mapping
#       was derived from the PalmOS SDK files Errorbase.h, NetMgr.h and InetMgr.h
#
#       This file expects to be included in CASL applications after
#       the CASLERR2.CPK file.
#
#
variables;
#
# Error code values base values for NetLib and InetLib
#
# base for NetLib (0x1200, NetLibErrorClass from Errorbase.h)
#
numeric cne_net_error_base  = 4608;
#
# base for INetLib (0x1400, InetLibErrorClass from Errorbase.h)
#
numeric cne_inet_error_base = 5120;

#
# Strings associated with each NetLib error code
#
numeric cne_number_of_net_error_strings = 82;

cne_net_error_strings[cne_number_of_net_error_strings] =

"NoError",                  # (netErrorClass | 0)
"AlreadyOpen",              # (netErrorClass | 1)
"NotOpen",                  # (netErrorClass | 2)
"StillOper",                # (netErrorClass | 3)
"ParamErr",                 # (netErrorClass | 4)
"NoMoreSockets",            # (netErrorClass | 5)
"OutOfResources",           # (netErrorClass | 6)
"OutOfMemory",              # (netErrorClass | 7)
"SocketNotOpen",            # (netErrorClass | 8)
"SocketBusy",               # (netErrorClass | 9)
"MessageTooBig",            # (netErrorClass | 10)
"SocketNotConnected ",      # (netErrorClass | 11)
"NoInterfaces",             # (netErrorClass | 12)
"BufTooSmall",              # (netErrorClass | 13)
"Unimplemented",            # (netErrorClass | 14)
"PortInUse",                # (netErrorClass | 15)
"QuietTimeNotElapsed",      # (netErrorClass | 16)
"Internal",                 # (netErrorClass | 17)
"Timeout",                  # (netErrorClass | 18)
"SocketAlreadyConnected ",  # (netErrorClass | 19)
"SocketClosedByRemote ",    # (netErrorClass | 20)
"OutOfCmdBlocks",           # (netErrorClass | 21)
"WrongSocketType",          # (netErrorClass | 22)
"SocketNotListening",       # (netErrorClass | 23)
"UnknownSetting",           # (netErrorClass | 24)
"InvalidSettingSize",       # (netErrorClass | 25)
"PrefNotFound",             # (netErrorClass | 26)
"InvalidInterface",         # (netErrorClass | 27)
"InterfaceNotFound",        # (netErrorClass | 28)
"TooManyInterfaces",        # (netErrorClass | 29)
"BufWrongSize",             # (netErrorClass | 30)
"UserCancel",               # (netErrorClass | 31)
"BadScript",                # (netErrorClass | 32)
"NoSocket",                 # (netErrorClass | 33)
"SocketRcvBufFull",         # (netErrorClass | 34)
"NoPendingConnect",         # (netErrorClass | 35)
"UnexpectedCmd",            # (netErrorClass | 36)
```

```
1385  "NoTCB",                      # (netErrorClass | 37)
1386  "NilRemoteWindowSize",        # (netErrorClass | 38)
1387  "NoTimerProc",                # (netErrorClass | 39)
1388  "SocketInputShutdown",        # (netErrorClass | 40)
1389  "CmdBlockNotCheckedOut",      # (netErrorClass | 41)
1390  "CmdNotDone",                 # (netErrorClass | 42)
1391  "UnknownProtocol",            # (netErrorClass | 43)
1392  "UnknownService",             # (netErrorClass | 44)
1393  "UnreachableDest",            # (netErrorClass | 45)
1394  "ReadOnlySetting",            # (netErrorClass | 46)
1395  "WouldBlock",                 # (netErrorClass | 47)
1396  "AlreadyInProgress",          # (netErrorClass | 48)
1397  "PPPTimeout",                 # (netErrorClass | 49)
1398  "PPPBroughtDown",             # (netErrorClass | 50)
1399  "AuthFailure",                # (netErrorClass | 51)
1400  "PPPAddressRefused",          # (netErrorClass | 52)
1401  "NameTooLong",                # (netErrorClass | 53)
1402  "DNSBadName",                 # (netErrorClass | 54)
1403  "DNSBadArgs",                 # (netErrorClass | 55)
1404  "DNSLabelTooLong",            # (netErrorClass | 56)
1405  "DNSAllocationFailure",       # (netErrorClass | 57)
1406  "DNSTimeout",                 # (netErrorClass | 58)
1407  "DNSUnreachable",             # (netErrorClass | 59)
1408  "DNSFormat",                  # (netErrorClass | 60)
1409  "DNSServerFailure",           # (netErrorClass | 61)
1410  "DNSNonexistantName",         # (netErrorClass | 62)
1411  "DNSNIY",                     # (netErrorClass | 63)
1412  "DNSRefused",                 # (netErrorClass | 64)
1413  "DNSImpossible",              # (netErrorClass | 65)
1414  "DNSNoRRS",                   # (netErrorClass | 66)
1415  "DNSAborted",                 # (netErrorClass | 67)
1416  "DNSBadProtocol",             # (netErrorClass | 68)
1417  "DNSTruncated",               # (netErrorClass | 69)
1418  "DNSNoRecursion",             # (netErrorClass | 70)
1419  "DNSIrrelevant",              # (netErrorClass | 71)
1420  "DNSNotInLocalCache",         # (netErrorClass | 72)
1421  "DNSNoPort",                  # (netErrorClass | 73)
1422  "IPCantFragment",             # (netErrorClass | 74)
1423  "IPNoRoute",                  # (netErrorClass | 75)
1424  "IPNoSrc",                    # (netErrorClass | 76)
1425  "IPNoDst",                    # (netErrorClass | 77)
1426  "IPPktOverflow",              # (netErrorClass | 78)
1427  "TooManyTCPConnections",      # (netErrorClass | 79)
1428  "NoDNSServers",               # (netErrorClass | 80)
1429  "InterfaceDown"               # (netErrorClass | 81)
1430  ;
1431
1432
1433  #
1434  # Strings associated with each InetLib error code
1435  #
1436  numeric cne_number_of_inet_error_strings = 96;
1437
1438  string cne_inet_error_strings[cne_number_of_inet_error_strings] =
1439
1440  "NoError",                          # InetLibErrorClass | 00
1441  "TooManyClients",                   # InetLibErrorClass | 01
1442  "HandleInvalid",                    # InetLibErrorClass | 02
1443  "ParamsInvalid",                    # InetLibErrorClass | 03
1444  "URLVersionInvalid",                # InetLibErrorClass | 04
1445  "URLBufTooSmall",                   # InetLibErrorClass | 05
1446  "URLInvalid",                       # InetLibErrorClass | 06
1447  "TooManySockets",                   # InetLibErrorClass | 07
1448  "NoRequestCreated",                 # InetLibErrorClass | 08
1449  "NotConnected",                     # InetLibErrorClass | 09
1450  "InvalidRequest",                   # InetLibErrorClass | 10
1451  "NeedTime",                         # InetLibErrorClass | 11
1452  "HostnameInvalid",                  # InetLibErrorClass | 12
1453  "InvalidPort",                      # InetLibErrorClass | 13
1454  "InvalidHostAddr",                  # InetLibErrorClass | 14
1455  "NilBuffer",                        # InetLibErrorClass | 15
```

```
1456  "ConnectTimeout",                    # InetLibErrorClass | 16
1457  "ResolveTimeout",                    # InetLibErrorClass | 17
1458  "SendReqTimeout",                    # InetLibErrorClass | 18
1459  "ReadTimeout",                       # InetLibErrorClass | 19
1460  "BufTooSmall",                       # InetLibErrorClass | 20
1461  "SchemeNotSupported",                # InetLibErrorClass | 21
1462  "InvalidResponse",                   # InetLibErrorClass | 22
1463  "Empty",
1464  "Empty",
1465  "SettingTooLarge",                   # InetLibErrorClass | 25
1466  "SettingSizeInvalid",                # InetLibErrorClass | 26
1467  "RequestTooLong",                    # InetLibErrorClass | 27
1468  "SettingNotImplemented",             # InetLibErrorClass | 28
1469  "ConfigNotFound",                    # InetLibErrorClass | 29
1470  "ConfigCantDelete",                  # InetLibErrorClass | 30
1471  "ConfigTooMany",                     # InetLibErrorClass | 31
1472  "ConfigBadName",                     # InetLibErrorClass | 32
1473  "ConfigNotAlias",                    # InetLibErrorClass | 33
1474  "ConfigCantPointToAlias",            # InetLibErrorClass | 34
1475  "ConfigEmpty",                       # InetLibErrorClass | 35
1476  "Empty",
1477  "ConfigAliasErr",                    # InetLibErrorClass | 37
1478  "NoWirelessInterface",               # InetLibErrorClass | 38
1479  "EncryptionNotAvail",                # InetLibErrorClass | 39
1480  "NeedRetryEncSeqNum",                # InetLibErrorClass | 40
1481  "NeedRetryEncPublicKey",             # InetLibErrorClass | 41
1482  "ResponseTooShort",                  # InetLibErrorClass | 42
1483  "MobitexIllegalOKHost",              # InetLibErrorClass | 43
1484  "MobitexIllegalBadHost",             # InetLibErrorClass | 44
1485  "HTTPBadRequest",                    # InetLibErrorClass | 45
1486  "HTTPUnauthorized",                  # InetLibErrorClass | 46
1487  "HTTPForbidden",                     # InetLibErrorClass | 47
1488  "HTTPNotFound",                      # InetLibErrorClass | 48
1489  "HTTPMethodNotAllowed",              # InetLibErrorClass | 49
1490  "HTTPNotAcceptable",                 # InetLibErrorClass | 50
1491  "HTTPProxyAuthRequired",             # InetLibErrorClass | 51
1492  "HTTPRequestTimeout",                # InetLibErrorClass | 52
1493  "HTTPConflict",                      # InetLibErrorClass | 53
1494  "HTTPGone",                          # InetLibErrorClass | 54
1495  "HTTPLengthRequired",                # InetLibErrorClass | 55
1496  "HTTPPreconditionFailed",            # InetLibErrorClass | 56
1497  "HTTPRequestTooLarge",               # InetLibErrorClass | 57
1498  "HTTPRequestURITooLong",             # InetLibErrorClass | 58
1499  "HTTPUnsupportedType",               # InetLibErrorClass | 59
1500  "HTTPServerError",                   # InetLibErrorClass | 60
1501  "CTPServerError",                    # InetLibErrorClass | 61
1502  "TypeNotCached",                     # InetLibErrorClass | 62
1503  "ErrCacheInvalid",                   # InetLibErrorClass | 63
1504  "URLDispatched",                     # InetLibErrorClass | 64
1505  "DatabaseNotFound",                  # InetLibErrorClass | 65
1506  "CTPMalformedRequest",               # InetLibErrorClass | 66
1507  "CTPUnknownCommand",                 # InetLibErrorClass | 67
1508  "CTPTruncated",                      # InetLibErrorClass | 68
1509  "CTPUnknownError",                   # InetLibErrorClass | 69
1510  "CTPProxyError",                     # InetLibErrorClass | 70
1511  "CTPSocketErr",                      # InetLibErrorClass | 71
1512  "CTPInvalidURL",                     # InetLibErrorClass | 72
1513  "CTPReferringPageOutOfDate",         # InetLibErrorClass | 73
1514  "CTPBadRequest",                     # InetLibErrorClass | 74
1515  "UNUSED",                            # InetLibErrorClass | 75
1516  "CTPMailServerDown",                 # InetLibErrorClass | 76
1517  "CTPHostNotFound",                   # InetLibErrorClass | 77
1518  "CTPContentInvalidTag",              # InetLibErrorClass | 78
1519  "CTPContentInternal",                # InetLibErrorClass | 79
1520  "CTPContentDataEnd",                 # InetLibErrorClass | 80
1521  "CTPContentResourceTooBig",          # InetLibErrorClass | 81
1522  "CTPContentNoNoFrames",              # InetLibErrorClass | 82
1523  "CTPContentUnsupportedContent",      # InetLibErrorClass | 83
1524  "CTPContentUnsupportedEncoding",     # InetLibErrorClass | 84
1525  "CTPContentBadForm",                 # InetLibErrorClass | 85
1526  "CTPContentBadFormMissingAction",    # InetLibErrorClass | 86
```

```
1527  "CTPContentBadFormMissingMethod",   # InetLibErrorClass | 87
1528  "CTPContentNoSourceData",           # InetLibErrorClass | 88
1529  "CTPContentBadImage",               # InetLibErrorClass | 89
1530  "CTPContentImageTooLarge",          # InetLibErrorClass | 90
1531  "MobitexErrorHandled",              # InetLibErrorClass | 91
1532  "ProxyDownBadHost",                 # InetLibErrorClass | 92
1533  "HostConnectionLost",               # InetLibErrorClass | 93
1534  "LinkNotFound"                      # InetLibErrorClass | 94
1535  ;
1536
1537  #
1538  # these networking error values are generated by the CASLrt or
1539  #  the CASLpro support library
1540  #
1541  numeric cne_socket_create      = -51;
1542  numeric cne_protocol_spec      = -52;
1543  numeric cne_missing_node       = -53;
1544  numeric cne_protocol_not_found = -54;
1545  numeric cne_host_lookup        = -55;
1546  numeric cne_no_connection      = -56;
1547  numeric cne_socket_closed      = -57;
1548
1549  cne_string = "";                    # CASL networking error string
1550
1551  end;
1552
1553  #
1554  # function to put appropriate error string into the variable "cne_string"
1555  #  for use by programs including this CASL package.
1556  #
1557  function cne_geterror as string;
1558  if errorcode > 0;
1559
1560  compile_if platform = "pilot";
1561
1562    if  (errorcode >= cne_net_error_base)
1563    and (errorcode < (cne_net_error_base + cne_number_of_net_error_strings) );
1564      cne_geterror = cne_net_error_strings[errorcode - cne_net_error_base];
1565    else_if (errorcode >= cne_inet_error_base)
1566    and (errorcode < (cne_inet_error_base + cne_number_of_inet_error_strings) );
1567      cne_geterror = cne_inet_error_strings[errorcode - cne_inet_error_base];
1568    else;
1569      cne_geterror = "OS error: " + string(errorcode, "");
1570    end_if;
1571
1572  compile_else;
1573
1574    cne_geterror = "OS error: " + string(errorcode, "");
1575
1576  compile_end_if;
1577
1578  else_if errorcode = 0;
1579    cne_geterror = "NoError";
1580  else_if errorcode = cne_socket_create;
1581    cne_geterror = "SocketCreatError";
1582  else_if errorcode = cne_protocol_spec;
1583    cne_geterror = "BadProtocol";
1584  else_if errorcode = cne_missing_node;
1585    cne_geterror = "MissingHost";
1586  else_if errorcode = cne_protocol_not_found;
1587    cne_geterror = "ProtocolNotFound";
1588  else_if errorcode = cne_host_lookup;
1589    cne_geterror = "HostNotFound";
1590  else_if errorcode = cne_no_connection;
1591    cne_geterror = "NoConnection";
1592  else_if errorcode = cne_socket_closed;
1593    cne_geterror = "SocketClosed";
1594  else;
1595    cne_geterror = ce_errors[-errorcode];
1596  end_if;
1597  end;
```

1598
1599

```
#
# CASL_Palm_Utils.cpk
#
# CASL interface definition to C functions to perform PalmOS operations
# not currently directly supported in CASL.
#
# This file defines the call interfaces to callable C functions that reside
# in the CASL_Palm_Utils library.  The function definitions are wrapped inside
# the EXTERNAL specification of CASL, indicating that code for the functions
# themselves is resident in the external library.
#
# Copyright(c) 2001 Feras Information Technologies
# Use of this code is permitted for any purpose, provided
#  you leave the FIT copyright notice, and you accept code
#  "AS IS" without warranty of any kind.
#

# ver 2.6.0.2, 7jan01, frank o'brien, caslsoft
#  support@caslsoft.com, www.caslsoft.com
# added supplementary usage comments and CASL_FixTimeBug function

# ver 2.6.0.1, 22oct98, joan feras, caslsoft
# original file

external "CASL_Palm_Utils";

# CASL_SelectDay
# Inputs are clear.
# When presented with dialog, user can tap OK or Cancel.
# If user taps OK, the return string is date in format "YYYYMMDD".
# If user taps Cancel, the return string is "" (null string).

function CASL_SelectDay    # display and manipulate the day pick dialog
  (
  string title,            #  title for the date picking form
  string day_string        #  string containing start date using format YYYYMMDD
  ) as string;             #  returns chosen date using same format


# CASL_SelectTime
# The 'times' input array must be preloaded with values that make sense.
#  Time format is HHMM, 24 h format string, examples: 6:15a="0615",
#  6:05p="1805", midnight="0000", noon="1200".
#  The start time element of array serves as default highlighted
#  value of time selector. Array must have at least 2 elements, even
#  if you want only 1 time, you could get some crashes if you pass
#  a 2nd null string or no 2nd element. If you pass 2 null strings,
#  the dialog won't show and the return value will be 0 (Cancel).
# The 'start_of_day' input is an integer numeric of what the time
#  dialog selector should start with. for example passing 6,
#  will have 6:00a as first visible time without needing scroll arrows,
#  18 will have 6:00p. It's smart enough to override this setting if
#  needed to show the inputed start 'times' value.
# When presented with dialog, user can tap OK or Cancel.
# If user taps OK, the return numeric is 1. This does not necessarily
#  mean that the time changed, it could still be default,
#  it only means OK has tapped. Returned 'times' elements are the new
#  selected times.
# If user taps Cancel, the return numeric is 0. It appears that the
#  passed 'times' elements are returned unchanged even if user had
#  different times selected before tapping cancel.
# With CASL_SelectTime, when return time is 00mm time, the actual
#  return value will be 0mm, when return time is 000m, the actual
#  return value will be 0m, this is bug in Palm API pick time dialog,
# for return times between 12 midnight (0000) and 12:09a (0009),
#  to get the correct 4 digit return time, hhmm, including any
#  preceeding 0's, use CASL_FixTimeBug

function CASL_SelectTime   # display and manipulate the time pick dialog
  (
  string  times[],         #  string array with start/end times at cells 0 and 1
```

```
1671    string  title,              #  title for the time picking form
1672    numeric start_of_day        #  top hour displayed in hour list
1673    ) as numeric;               #  returns 1 if user taps ok, 0 otherwise
1674
1675  # CASL_SysGetRomToken
1676  ### John, you may want to add some comments here,
1677  ###  not sure if doesn't work on Palm VII??
1678
1679                                # get the serial number of the PalmIII ROM (if possible)
1680                                #
1681  function CASL_SysGetRomToken as string;
1682
1683  end_external;
1684
1685  # CASL_FixTimeBug
1686  # given time in format hhmm, or 0mm, or 0m,
1687  #  such as an element of the 'times' array from CASL_SelectTime,
1688  # returns hhmm,
1689  #  hhmm -> hhmm
1690  #  0mm -> 00mm
1691  #  0m -> 000m
1692
1693  function CASL_FixTimeBug(string time) as string;
1694      if length(time)=4;
1695          CASL_FixTimeBug=time;
1696      else;
1697          if length(time)=3;
1698              CASL_FixTimeBug="0"+time;
1699          else;
1700              CASL_FixTimeBug="0"+left(time,1)+"0"+right(time,1);
1701          end_if;
1702      end_if;
1703  end;
1704
1705
```

```
#
# CASLER26.CPK - Package containing CASL 2.0 error codes/functions
#
# This file is a CASL package intended for inclusion in CASL programs
# that will make use of the interpreter error codes and their associated
# strings.
#
# NOTE: this file is designed specifically for use with CASL version 2.0
#       due to its usage function return values, and the "bad number"
#       error code.
#
# Copyright(c) 2001 Feras Information Technologies
# Use of this code is permitted for any purpose, provided
#  you leave the FIT copyright notice, and you accept code
#  "AS IS" without warranty of any kind.
#

# Modifications by Scott Smith, March 1999 to include ce_no_error and to
# remove reference to ce_string variable that is no longer used/needed.

variables;

  #
  # Error code values
  #
  ce_no_error = 0;
  ce_div_0 = -1;
  ce_array = -2;
  ce_not_open = -3;
  ce_fld_mis = -4;
  ce_eof = -5;
  ce_fmode = -6;
  ce_file_nf = -7;
  ce_file_del = -8;
  ce_serial_open = -9;
  ce_serial_comm = -10;
  ce_file_open = -11;
  ce_rec_not_found = -12;
  ce_put_record_error = -13;
  ce_file_delete_error = -14;
  ce_bad_number_error = -15;

  #
  # Strings associated with each error code
  #

  ce_errors[16] = "No Error",              "Divide by 0",
                  "Invalid array index",   "File not open",
                  "DB field mismatch",     "File EOF",
                  "Invalid file mode",     "File not found",
                  "Record deleted",        "Serial port error",
                  "Serial comm error",     "File already open",
                  "Record not found",      "Put record error",
                  "Delete record error",   "Bad Number Error";

  #
  # CASL error string (no longer used)
  #

  # ce_string = "";

end;

#
# function to put appropriate error string into the variable "ce_string"
#  for use by programs including this CASL package.
#

function ce_geterror as string;

  variables;
```

```
1777      numeric temp;
1778    end;
1779
1780    if errorcode > 0;
1781      ce_geterror = "OS error: " + string(errorcode, "");
1782    else;
1783      ce_geterror = ce_errors[-errorcode];
1784    end_if;
1785
1786    if false();
1787      # just to prevent "var not used" warning messages
1788      temp = ce_no_error + ce_div_0 + ce_array + ce_not_open + ce_fld_mis
1789           + ce_eof + ce_fmode + ce_file_nf + ce_file_del + ce_serial_open
1790           + ce_serial_comm + ce_file_open + ce_rec_not_found
1791           + ce_put_record_error + ce_file_delete_error + ce_bad_number_error;
1792    end_if;
1793
1794  end;
1795
1796
```

```
1797  #
1798  # dba_DBAccess.cpk,
1799  #  generic dbfile functions
1800  #
1801  # Copyright(c) 2001 Feras Information Technologies
1802  # Use of this code is permitted for any purpose, provided
1803  #  you leave the FIT copyright notice, and you accept code
1804  #  "AS IS" without warranty of any kind.
1805  #
1806
1807  # ver 3.1.0.1, 7jan01, frank o'brien, caslsoft
1808  #  fobrien@caslsoft.com, www.caslsoft.com
1809
1810  # this file contains all basic functions needed to work with
1811  #  with dbfiles.
1812  # functions are provided to: create/open db, existing
1813  #  record check and select, write record, read exact record,
1814  #  read bestfit record, remove record.
1815  # all these functions return true/false values for
1816  #  success.  this makes if easy to do errorchecking
1817  #  and leaves error messages to the programmer,
1818  # these functions are defined without a return value:
1819  #  close db, delete db.
1820
1821  # depending on dbfile usage, it's sometimes useful to fill an array with
1822  #  all key field strings in the dbfile to ease record selection,
1823  #  no generic function is provided in this package, as the bound
1824  #  field variables are specific to each dbfile.  this task is left
1825  #  to programmer.  normally this is done similar to this:
1826  #
1827  # i=0;
1828  # errorcode=0;
1829  # get_fields dbData;
1830  # while errorcode=0;
1831  #   sKeyList[1]=sKey;
1832  #   i=i+1;
1833  #   get_fields dbData;
1834  # end_while;
1835  # if errorcode<>-5; # eof
1836  #   # show error
1837  # else;
1838  #   i is 1-based record count, all keys are in sKeyList array
1839  # end_if;
1840  #
1841  # for dbfile that is used on both palm and windows platforms,
1842  #  with merge sync_pref, and when same record is changed on
1843  #  each platform between hotsync, can get duplicate records, and/or
1844  #  number of records exceeding max allowed (if there is a limit),
1845  #  sometimes may want these checks in fill loop too.
1846
1847  # create/open db
1848
1849  # given dbfile object, and filename,
1850  # sideeffects: if db does not exist, db will
1851  #  be created with filename and record structure
1852  #  defined by bound field variables, if db
1853  #  exists, opens db with filename, errorcode has error status,
1854  # if desired, use CASL file function "Exists" to check is
1855  #  db file exists prior to call,
1856  # returns true/false with success
1857
1858  function dba_ffOpenDB(object dbobj,string dbname) as numeric;
1859      errorcode=0;
1860      open dbobj,dbname;
1861      dba_ffOpenDB=errorcode=0;
1862  end;
1863
1864  # given dbfile object, and key field data string,
1865  # side effect: db pointer set to matching record if found,
1866  #  or record right after in sort order if not found,
1867  #  errorcode has error status
```

```
# returns true if key is unique
# returns false if key was found, or unexpected error

function dba_ffRecordExists(object dbobj,numeric key) as numeric;
    errorcode=0;
    search dbobj,key;
    dba_ffRecordExists=errorcode=0;
end;

# write record

# db bound variables already have field data
# given dbfile object, and key field data string,
# side effect: if key is new insert new record, if key
#  already exists writes new data for existing record,
#  errorcode has error status
# returns true/false on success

function dba_ffWriteRecord(object dbobj,numeric key) as numeric;
    if dba_ffRecordExists(dbobj,key);
        # write existing record
        errorcode=0;
        put_fields dbobj;
    else;
        # add new record
        errorcode=0;
        insert dbobj;
    end_if;
    dba_ffWriteRecord=errorcode=0;
end;

# read record, 2 versions

# given dbfile object, and key field data string,
# side effect: fills db bound variables with field data
#  of record exactly matching key field data string,
#  errorcode has error status
# returns true/false on success

function dba_ffReadExactRecord(object dbobj,numeric key) as numeric;
    if dba_ffRecordExists(dbobj,key);
        # found
        errorcode=0;
        get_fields dbobj;
        dba_ffReadExactRecord=errorcode=0;
    else;
        # not found
        dba_ffReadExactRecord=false;
    end_if;
end;


# given dbfile object, and key field data string,
# side effect: fills db bound variables with field
#  data from matching record if found,
#  or record right after in sort order if not found,
#  errorcode has error status
# returns true/false on success

function dba_ffReadBestFitRecord(object dbobj,numeric key) as numeric;
    if dba_ffRecordExists(dbobj,key);
        # found
        errorcode=0;
        get_fields dbobj;
        dba_ffReadBestFitRecord=errorcode=0;
    else;
        # not found, read closest fit
        errorcode=0;
        get_fields dbobj;
        dba_ffReadBestFitRecord=errorcode=0;
    end_if;
```

```
end;


# given dbfile object, and a record number,
# side effect: fills db bound variables with field data
#  errorcode has error status
# returns true/false on success
function dba_ffReadRecordFromRecNo(object dbobj, numeric recNo) as numeric;
    errorcode=0;
    seek_start dbobj, recNo;

    if errorcode=0;
        # found
        errorcode=0;
        get_fields dbobj;
        dba_ffReadRecordFromRecNo=errorcode=0;
    else;
        # not found
        dba_ffReadRecordFromRecNo=false;
    end_if;
end;


# remove record

# given dbfile object, and key field data string,
# side effect: removes record with matching key field
#  data string from db, errorcode has error status,
# returns true/false on success, returns true if doesn't exist

function dba_ffRemoveRecord(object dbobj,numeric key) as numeric;
    if dba_ffRecordExists(dbobj,key);
        errorcode=0;
        remove dbobj;
        dba_ffRemoveRecord=errorcode=0;
    else;
        dba_ffRemoveRecord=true;
    end_if;
end;

# close db

# given dbfile object,
# sideeffect: closes dbfile, errorcode has error status,
# it's assumed programmer will normally not be able to do
#  anything about failure, so there's no return value,
#  if programmer is interested in error status, the
#  system's errorcode value can be queried after call,
#  in this way, programmer could add a "retry/abort" option.

function dba_CloseDB(object dbobj);
    errorcode=0;
    close dbobj;
end;

# delete db

# given dbfile object, and filename,
# sideeffect: deletes dbfile, dbfile will be closed
#  before delete if it's open, errorcode has error status,
# it's assumed programmer will normally not be able to do
#  anything about failure, so there's no return value,
#  if programmer is interested in error status, the
#  system's errorcode value can be queried after call,
#  in this way, programmer could add a "retry/abort" option.

function dba_DeleteDB(object dbobj,string dbname);
    errorcode=0;
    if Exists(dbname);
        dba_CloseDB(dbobj); # can't delete open file
        delete dbname;
```

```
2010      end_if;
2011  end;
2012
2013
```

```
#
# fdt_FormatDateTime.cpk,
#  gregorian date, weekday, and time formatting functions
#
# Copyright(c) 2001 Feras Information Technologies
# Use of this code is permitted for any purpose, provided
#  you leave the FIT copyright notice, and you accept code
#  "AS IS" without warranty of any kind.
#

#  see also:
#   fdt_JulianDateTime.cpk for julian date and time funcitons,
#   gdt_GregorianDateTime.cpk for gregorian date and time functions

# ver 3.1.0.2, 21jan01, frank o'brien, caslsoft
#  support@caslsoft.com, www.caslsoft.com

# given Gregorian date in format, YYYYMMDD,
#  where YYYY=-4900 to 9999, MM=month=01-12, DD=day=01-31,
#  and dateformat string in form "dd mmm yy", or "dd mmm yyyy",
#  or "mm/dd/yy" or "dd.mm.yy",
#  dateformat string is case-insensitive,
# returns Gregorian date in format specified, where
#  dd=days without leading zero
#  mm=month without leading zero
#  mmm=month of form Jan-Dec,
#  yy=last 2 digits of year with leading zero
#  yyyy=year
# if dateformat string is unrecognized, date is returned in format
#  "dd mmm yy",
# no error checking

function fdt_sfFormatDate(string gd,string dateformat) as string;
    variables;
        numeric y;
        numeric m;
        numeric d;
    end;
    y=value(left(gd,length(gd)-4));
    m=value(mid(gd,length(gd)-4,2));
    d=value(right(gd,2));
    if(lower(dateformat)="mm/dd/yyyy");
        # format of form 3/17/2001
        fdt_sfFormatDate=
        string(m,"")+"/"+
        string(d,"")+"/"+
        string(y,"");
    else;
        if lower(dateformat)="mm/dd/yy";
            # format of form 3/17/01
            fdt_sfFormatDate=
             string(m,"")+"/"+
             string(d,"")+"/"+
             string(y%100,"0#");
        else;
            if lower(dateformat)="dd.mm.yy";
                # format of form 17.3.01
                fdt_sfFormatDate=
                 string(d,"")+". "+
                 string(m,"")+". "+
                 string(y%100,"0#");
            else;
                if lower(dateformat)="dd mmm yyyy";
                    # format of form 17 Mar 2001
                    fdt_sfFormatDate=
                     string(d,"")+" "+
                     mid("JanFebMarAprMayJunJulAugSepOctNovDec",(m-1)*3,3)+" "+
                     string(y,"");
                else;
                    # format of form 17 Mar 01, or unrecognized format
                    fdt_sfFormatDate=
```

```
                        string(d,"")+" "+
                        mid("JanFebMarAprMayJunJulAugSepOctNovDec",(m-1)*3,3)+" "+
                        string(y%100,"0#");
                end_if;
            end_if;
        end_if;
    end_if;
end;

# given Gregorian time in format, HHmm, 24 h format.
#  and timeformat string in form "hh:mm 24", "hh:mm 12", or "hh.mm",
#  timeformat string is case-insensitive,
# returns time in specified format where,
#  hh=hours with leading zero, see also 24 and 12
#  mm=minutes with leading zero
#  24=hours are in modulo 24
#  12=hours are in modulo 12, suffix is "am" if hours=0-11 in modulo 24,
#   and "pm" if hours=12-23 in modulo 24
# if timeformat string is unrecognized, time is returned in format
#  "hh:mm 24",
# no error checking

function fdt_sfFormatTime(string gt,string timeformat) as string;
    variables;
        numeric h;
        numeric mn;
    end;
    h=value(left(gt,2));
    mn=value(mid(gt,2,2));
    if lower(timeformat)="hh.mm";
        # 24 h format, decimal separator, good for led font
        fdt_sfFormatTime=string(h,"0#")+". "+string(mn,"0#");
    else;
        if lower(timeformat)="hh:mm 12";
            # am/pm format
            if h/12<1;
                # am
                if h=0;
                    fdt_sfFormatTime="12:"+string(mn,"0#")+" am";
                else;
                    fdt_sfFormatTime=string(h,"0#")+":"+
                     string(mn,"0#")+" am";
                end_if;
            else;
                # pm
                if h=12;
                    fdt_sfFormatTime="12:"+string(mn,"0#")+" pm";
                else;
                    fdt_sfFormatTime=string(h%12,"0#")+": +
                     string(mn,"0#")+" pm";
                end_if;
            end_if;
        else;
            # 24 hour format with colon separator, or unrecognized format
            fdt_sfFormatTime=string(h,"0#")+":"+string(mn,"0#");
        end_if;
    end_if;
end;

# commented out because OClock sample app would exceed demo limit,
# change false to true if you have registered version, or not
#  using with OClock sample app

compile_if false;

# given weekday, 0=mon, ... 6=sun, and wdformat string in form
#  of "short" or "long", wdformat is case-insensitive,
# returns string of weekday where,
#  if "long", returns "Monday", ... "Sunday"
#  if "short" or unrecognized, returns "Mon", ... "Sun"
# no error checking (if wd out of range, should return
```

```
2156 #  null string, but not fully tested)
2157
2158 function fdt_sfFormatWeekday(numeric wd,string wdformat) as string;
2159     if lower(wdformat)="long";
2160         # long format
2161         variables;
2162             string days[7]="Monday","Tuesday","Wednesday","Thursday",
2163              "Friday","Saturday","Sunday";
2164         end;
2165         if wd<0 or wd>6;
2166             fdt_sfFormatWeekday="";
2167         else;
2168             fdt_sfFormatWeekday=days[wd];
2169         end_if;
2170     else;
2171         # short format or unrecognized format
2172         fdt_sfFormatWeekday=mid("MonTueWedThuFriSatSun",wd*3,3);
2173     end_if;
2174 end;
2175
2176 compile_end_if;
2177
2178
```

```
#
# jdt_JulianDateTime.cpk,
# julian date, weekday, and time functions,
#
# Copyright(c) 2001 Feras Information Technologies
# Use of this code is permitted for any purpose, provided
#  you leave the FIT copyright notice, and you accept code
#  "AS IS" without warranty of any kind.
#


#  see also:
#   gdt_GregorianDateTime.cpk for gregorian date and time functions
#   fdt_FormatDateTime.cpk for gregorian formatting functions

# Algorithm for converting between Gregorian dates and
#  Julian Dates presented in 1968 in a letter to the editor of
#  Communications of the ACM (CACM, volume 11, number 10,
#  October 1968, p.657) by Henry F. Fliegel and
#  Thomas C. Van Flandern
#
# Additional information from Peter Meyer at http://hermetic.magnet.ch/
#
# Astronomers and chronologists use a system of numbering days
# called Julian days. The temporal sequence of days is mapped
# onto the sequence of integers, -2, -1, 0, 1, 2, 3, etc. This
# makes it easy to determine the number of days between two days
# (just subtract one Julian day number from the other).
#
# Do not confuse Julian days with the Julian Calender.  Julian days
# are a serial date system so all Gregorian dates have a seqencial
# ordering to faciliate math involving Gregorian dates

# ver 3.1.0.1, 7jan01, frank o'brien, caslsoft
#  support@caslsoft.com, www.caslsoft.com

# given Gregorian date in format, YYYYMMDD,
#  where YYYY=-4900 to 9999, MM=month=01 to 12, DD=day=01 to 31,
# returns the Julian Days, a serial date,
# no error checking,
#  handles out of range value as a rollover type thing
# the julian date defaults to midnight (=.0)
# no error checking

function jdt_nfJulianDate(string gd) as numeric;
    variables;
        numeric y;
        numeric m;
        numeric d;
    end;
    y=value(left(gd,length(gd)-4));
    m=value(mid(gd,length(gd)-4,2));
    d=value(right(gd,2));
    jdt_nfJulianDate=int((1461*(y+4800+int((m-14)/12)))/4)+
     int((367*(m-2-12*(int((m-14)/12))))/12)-
     int((3*int((y+4900+int((m-14)/12))/100))/4)+
     d-32075;
end;

# given Gregorian time in format, HHmm, 24 h format,
#  where HH=hours=00 to 23, and mm==minutes=00 to 59,
# returns decimal eqivalent of number of days
#  midnight=0.00, noon=0.5, 6:00a=6/24=0.25,
#  6:45p=18/24+45/60/24=a little more than 0.75
# no error checking

function jdt_nfJulianTime(string gt) as numeric;
    variables;
        numeric h;
        numeric mn;
    end;
    h=value(left(gt,2));
```

```
2250      mn=value(mid(gt,2,2));
2251      jdt_nfJulianTime=(h+(mn/60))/24;
2252  end;
2253
2254  # commented out because DaysBetween and OClock sample apps
2255  #  would exceed demo limit, change false to true if you have
2256  #  registered version, or not using with DaysBetween and
2257  #  OClock sample app
2258
2259  compile_if false;
2260
2261  # given Gregorian date in format, YYYYMMDD,
2262  #  where YYYY=year=-4900 to 9999, MM=month=01 to 12, DD=day=01 to 31,
2263  # returns modulo 7 of julian date of gregorian date where,
2264  #  0=mon, ... 6=sun
2265  # no error checking
2266
2267  function jdt_nfWeekday(string gd) as numeric;
2268      jdt_nfWeekday=jdt_nfJulianDate(gd)%7;
2269  end;
2270
2271  # given Gregorian date in format, YYYYMMDD,
2272  #  where YYYY=year=-4900 to 9999, MM=month=01 to 12, DD=day=01 to 31,
2273  # returns true if inputed gregorian date fits within
2274  #  limits given above, returns false otherwise
2275
2276  function jdt_ffLegalDate(string gd) as numeric;
2277      variables;
2278          string y;
2279          string m;
2280          string d;
2281      end;
2282      y=left(gd,length(gd)-4);
2283      m=mid(gd,length(gd)-4,2);
2284      d=right(gd,2);
2285      jdt_ffLegalDate=(y="0000" or (value(y)>=-4900 and value(y)<=9999 and
2286       value(y)<>0)) and (value(m)>=1 and value(m)<=12) and
2287       (value(d)>=1 and value(d)<=31);
2288  end;
2289
2290  # given Gregorian time in format, HHmm, 24 h format,
2291  #  where HH=hours=00 to 23, and mn=-minutes=00 to 59,
2292  # returns true if inputed gregorian time fits within
2293  #  limits given above, returns false otherwise
2294
2295  function jdt_ffLegalTime(string gt) as numeric;
2296      variables;
2297          string h;
2298          string mn;
2299      end;
2300      h=left(gt,2);
2301      mn=mid(gt,2,2);
2302      jdt_ffLegalTime=(h="00" or (value(h)>0 and value(h)<24)) and
2303       (mn="00" or (value(mn)>0 and value(mn)<60));
2304  end;
2305
2306  compile_end_if;
2307
2308
```

```
#
# PIDS_date_lib.cpk
#
################################################################################
#
# PIDS (tm)
# Pharmacy Intervention Documentation System
# Copyright © 2002  Integrated Software Systems, L.L.C.
#
################################################################################
#
# Version:  1.0
# Date:     1/7/2002
# Author:   Todd Viegut & Brad Tice
#
#
# Function to handle retrieving the intervention date.
#
include "fdt_FormatDateTime.cpk";
include "jdt_JulianDateTime.cpk";
include "CASL_Palm_Utils.cpk";
#
function GetCurrentInterventionDate as string;
    variables;
        string sSelectedDate;
    end;
    sSelectedDate = string(year(),"0###") + string(month(),"0#") + string(day(),"0#");
    GetCurrentInterventionDate = fdt_sfFormatDate(sSelectedDate,"mm/dd/yy");
end;

function GetInterventionDate(string sDefaultDate) as string;
    compile_if platform = "windows";
        variables;
            numeric mb;
        end;
        mb = message_box(2,"Warning","Not intended to run on windows", "", "", "OX");
        GetInterventionDate = "";
    compile_else;
        variables;
            string sFormat;
            string sSelectedDate;
        end;
        #
        sFormat = string(year(),"0###") + string(month(),"0#") + string(day(),"0#");
        #
        # running on palm, use external library for date and time
        # get new date with date pick dialog
        #
        sSelectedDate = CASL_SelectDay("Intervention Date", sFormat);
        #sSelectedDate = "20020101";
        if (sSelectedDate <> "");
            GetInterventionDate = fdt_sfFormatDate(sSelectedDate, "mm/dd/yy");
        else;
            GetInterventionDate = sDefaultDate;
        end_if;
    compile_end_if;
end;
```

```
#
# PIDS_db_files.cpk
#
################################################################################
#
# PIDS (tm)
# Pharmacy Intervention Documentation System
# Copyright © 2002  Integrated Software Systems, L.L.C.
#
################################################################################
#
# Version:  1.0
# Date:     12/22/2001
# Author:   Todd Viegut & Brad Tice
#
#
# Database related variable definitions:
#
variables;

    #
    #---------------------------------
    # db fields for preferences...
    #---------------------------------
    #
    numeric nDBPrefsKeyField;
    numeric fDBPrefsEnableSpeedEntryField;
    string sDBPrefsWirelessURLField;
    string sDBPrefsLocationIDField;
    string sDBPrefsRotationIDField;
    string sDBPrefsUserIDField;

    #
    #---------------------------------
    # db fields for PIDS...
    #---------------------------------
    #
    numeric nDBPIDSKeyField;
    numeric nDBSynced;
    #
    string sDBRotationID;
    string sDBLocationID;
    string sDBInterventionDate;
    string sDBUserID;
    string sDBPrescribedDrug;
    string sDBDisease;
    string sDBPrescriber;
    #
    numeric nDBDrugRelatedProblemField;
    numeric nDBRXTypeField;
    numeric nDBPatientRiskField;
    numeric nDBProblemTypeField;
    numeric nDBRecommendationField;
    numeric nDBActivityField;
    numeric nDBResultField;
    numeric nDBTimeField;
    #
    string sDBPatientIDField;
    string sDBAllergiesField;
    string sDBMedicalHistoryField;
    string sDBFamilyHistoryField;
    string sDBSocialHistoryField;
    #
    string sDBSOAPSubjectiveField;
    string sDBSOAPObjectiveField;
    string sDBSOAPAssessmentField;
    string sDBSOAPPlanField;
    string sDBSOAPFollowUpField;
    #
    string sDBLabValues;
    numeric nDBDOIScore;
```

```
2439
2440      #
2441      #-----------------------------------
2442      # db fields for PIDS XML database...
2443      #-----------------------------------
2444      #
2445      numeric nDBXMLPIDSKeyField;
2446      numeric nDBXMLSynced;
2447      #
2448      string sDBXMLIntervention;
2449  end;
2450
2451  #
2452  #-------------------------------------------------
2453  #   DATABASE OBJECT DEFINITIONS
2454  #-------------------------------------------------
2455  #
2456  dbfile dbPids;
2457
2458      field nDBPIDSKeyField; # This is the record "key" index
2459      field nDBSynced;
2460      #
2461      field sDBRotationID;
2462      field sDBLocationID;
2463      field sDBInterventionDate;
2464      field sDBUserID;
2465      field sDBPrescribedDrug;
2466      field sDBDisease;
2467      field sDBPrescriber;
2468      #
2469      field nDBDrugRelatedProblemField;
2470      field nDBRXTypeField;
2471      field nDBPatientRiskField;
2472      field nDBProblemTypeField;
2473      field nDBRecommendationField;
2474      field nDBActivityField;
2475      field nDBResultField;
2476      field nDBTimeField;
2477      #
2478      field sDBPatientIDField;
2479      field sDBAllergiesField;
2480      field sDBMedicalHistoryField;
2481      field sDBFamilyHistoryField;
2482      field sDBSocialHistoryField;
2483      #
2484      field sDBSOAPSubjectiveField;
2485      field sDBSOAPObjectiveField;
2486      field sDBSOAPAssessmentField;
2487      field sDBSOAPPlanField;
2488      field sDBSOAPFollowUpField;
2489      #
2490      field sDBLabValues;
2491      #
2492      field nDBDOIScore;
2493
2494      sync_pref pda_to_pc;
2495
2496  end;
2497
2498  dbfile dbPidsXML;
2499
2500      field nDBXMLPIDSKeyField; # This is the record "key" index
2501      field nDBXMLSynced;
2502      field sDBXMLIntervention;
2503
2504      sync_pref pda_to_pc;
2505
2506  end;
2507
2508  dbfile dbPidsPrefs;
2509
```

```
2510      field nDBPrefsKeyField;
2511      field fDBPrefsEnableSpeedEntryField;
2512      field sDBPrefsWirelessURLField;
2513      field sDBPrefsLocationIDField;
2514      field sDBPrefsRotationIDField;
2515      field sDBPrefsUserIDField;
2516
2517      sync_pref pda_to_pc;
2518
2519  end;
2520
2521
```

```
2522  #
2523  # PIDS_error_handling.cpk
2524  #
2525  ##############################################################################
2526  #
2527  # PIDS (tm)
2528  # Pharmacy Intervention Documentation System
2529  # Copyright © 2002  Integrated Software Systems, L.L.C.
2530  #
2531  ##############################################################################
2532  #
2533  # Version:  1.0
2534  # Date:     12/23/2001
2535  # Author:   Todd Viegut & Brad Tice
2536  #
2537  #
2538  # Function for displaying a standardized message box for PIDS errors.
2539  #
2540  function ErrorMsg(string sMessage);
2541     variables;
2542          numeric nResult;
2543     end;
2544     nResult = message_box(1, "PIDS Error", sMessage + " [" + ce_geterror + "]", "", "Ok",
       "");
2545  end;
2546
2547
```

```
#
# PIDS_misc.cpk
#
################################################################################
#
# PIDS (tm)
# Pharmacy Intervention Documentation System
# Copyright © 2002  Integrated Software Systems, L.L.C.
#
################################################################################
#
# Version:  1.0
# Date:     12/14/2001
# Author:   Todd Viegut & Brad Tice
#
#
# Miscellaneous functions.
#


#
#-------------------------------------------------
# Automated data entry functions...
#-------------------------------------------------
#
function GetDRPTypeIndex(numeric nDrugRelatedProblemSelection) as numeric;
    variables;
        numeric fContinue;
        numeric nIndex;
    end;
    nIndex = 1;
    #
    fContinue = true;
    while ((nIndex < nTotalDRPFrames) and (fContinue));
        if ((nDrugRelatedProblemSelection >= nDRPSelectionListOffsets[nIndex]) and
        (nDrugRelatedProblemSelection < nDRPSelectionListOffsets[nIndex + 1]));
            fContinue = false;
        else;
            nIndex = nIndex + 1;
        end_if;
    end_while;
    #
    GetDRPTypeIndex = nIndex;
end;


function GetDRPTypeSubIndex(numeric nDrugRelatedProblemSelection) as numeric;
    variables;
        numeric nIndex;
    end;
    nIndex = GetDRPTypeIndex(nDrugRelatedProblemSelection);
    GetDRPTypeSubIndex = nDrugRelatedProblemSelection - nDRPSelectionListOffsets[nIndex];
end;

function GetDrugRelatedProblemData(numeric nDrugRelatedProblemSelection) as string;
    variables;
        object objSelector;
        numeric nIndex;
        numeric nSubIndex;
    end;
    nIndex = GetDRPTypeIndex(nDrugRelatedProblemSelection);
    nSubIndex = GetDRPTypeSubIndex(nDrugRelatedProblemSelection);
    objSelector = objDRPSelectors[nIndex - 1]; # This array is 0 based...
    #
    GetDrugRelatedProblemData = "DRUG RELATED PROBLEM:" + char(10) +
    drug_related_problem_list[nIndex] + " - " + objSelector.display;
end;

function GetPatientInfoData() as string;
    GetPatientInfoData = ""; # Must initialize to empty string...
    #
    # Allergies:
```

```
2617      if (length(txAllergies.display) > 0);
2618          GetPatientInfoData = "ALLERGIES:" + char(10);
2619          GetPatientInfoData = GetPatientInfoData + txAllergies.display;
2620      end_if;
2621      #
2622      # Medication History:
2623      if (length(txMedHistory.display) > 0);
2624          if (length(GetPatientInfoData) > 0);
2625              GetPatientInfoData = GetPatientInfoData + char(10) + char(10) + "MED HISTORY:"
                    + char(10);
2626          else;
2627              GetPatientInfoData = "MED HISTORY:" + char(10);
2628          end_if;
2629          GetPatientInfoData = GetPatientInfoData + txMedHistory.display;
2630      end_if;
2631      #
2632      # Family History:
2633      if (length(txFamilyHistory.display) > 0);
2634          if (length(GetPatientInfoData) > 0);
2635              GetPatientInfoData = GetPatientInfoData + char(10) + char(10) + "FAMILY
                    HISTORY:" + char(10);
2636          else;
2637              GetPatientInfoData = "FAMILY HISTORY:" + char(10);
2638          end_if;
2639          GetPatientInfoData = GetPatientInfoData + txFamilyHistory.display;
2640      end_if;
2641      #
2642      # Social History:
2643      if (length(txSocialHistory.display) > 0);
2644          if (length(GetPatientInfoData) > 0);
2645              GetPatientInfoData = GetPatientInfoData + char(10) + char(10) + "SOCIAL
                    HISTORY:" + char(10);
2646          else;
2647              GetPatientInfoData = "SOCIAL HISTORY:" + char(10);
2648          end_if;
2649          GetPatientInfoData = GetPatientInfoData + txSocialHistory.display;
2650      end_if;
2651  end;
2652
2653  function GetLabValuesData(string sInitialString) as string;
2654      variables;
2655          numeric nLoopIndex;
2656      end;
2657      GetLabValuesData = sInitialString;
2658      for nLoopIndex = 0, nLoopIndex < nLabValuesCreated;
2659          if (length(GetLabValuesData) > 0);
2660              GetLabValuesData = GetLabValuesData + char(10) +
                    completed_lab_values[nLoopIndex];
2661          else;
2662              GetLabValuesData = GetLabValuesData + completed_lab_values[nLoopIndex];
2663          end_if;
2664      next nLoopIndex + 1;
2665  end;
2666
2667  #
2668  #-------------------------------------------------
2669  # DOI Score(tm) calculation...
2670  #-------------------------------------------------
2671  #
2672  function CalculateDOIScore as numeric;
2673      variables;
2674          object objSelector;
2675          numeric nFrameIndex;
2676          numeric nWeightIndex;
2677          numeric nSelectedIndex;
2678      end;
2679      #
2680      CalculateDOIScore = 0;
2681      #
2682      for nFrameIndex = 2, nFrameIndex <= 9;
2683          objSelector = objInterventionSelectors[nFrameIndex];
```

```
2684            nSelectedIndex = objSelector.selected;
2685            #
2686            if nFrameIndex = 2;
2687                objSelector = objDRPSelectors[nDRPSelected - 1];
2688                nSelectedIndex = objSelector.selected;
2689                nWeightIndex = (nDRPSelectionListOffsets[nDRPSelected] + nSelectedIndex);
2690                CalculateDOIScore = CalculateDOIScore + nDRPWeights[nWeightIndex];
2691            else_if nFrameIndex = 3;
2692                CalculateDOIScore = CalculateDOIScore + nRXTypeWeights[nSelectedIndex];
2693            else_if nFrameIndex = 4;
2694                CalculateDOIScore = CalculateDOIScore + nPatientRiskWeights[nSelectedIndex];
2695            else_if nFrameIndex = 5;
2696                CalculateDOIScore = CalculateDOIScore + nProblemTypeWeights[nSelectedIndex];
2697            else_if nFrameIndex = 6;
2698                CalculateDOIScore = CalculateDOIScore +
                    nRecommendationWeights[nSelectedIndex];
2699            else_if nFrameIndex = 7;
2700                CalculateDOIScore = CalculateDOIScore + nActivityWeights[nSelectedIndex];
2701            else_if nFrameIndex = 8;
2702                CalculateDOIScore = CalculateDOIScore + nResultWeights[nSelectedIndex];
2703            else_if nFrameIndex = 9;
2704                CalculateDOIScore = CalculateDOIScore + nTimeWeights[nSelectedIndex];
2705            end_if;
2706        next nFrameIndex + 1;
2707    end;
2708
2709    #
2710    #-------------------------------------------------
2711    # Functions invoked for frame navigation using
2712    # BACK and NEXT buttons...
2713    #-------------------------------------------------
2714    #
2715    function HideFrame;
2716        if (nCurrentFrame >= 0) and (nCurrentFrame < nTotalFrames);
2717            if (nContainsSelector[nCurrentFrame]);
2718                hide objInterventionSelectors[nCurrentFrame];
2719            end_if;
2720            hide objPIDSFrames[nCurrentFrame];
2721        end_if;
2722    end;
2723
2724    function ShowFrame;
2725        if ((nCurrentFrame >= 0) and (nCurrentFrame < nTotalFrames));
2726            if objPIDSFrames[nCurrentFrame] = save_intervention_frame;
2727                txDOIScore.display = string(CalculateDOIScore, "#");
2728            end_if;
2729            if (nContainsSelector[nCurrentFrame]);
2730                show objInterventionSelectors[nCurrentFrame];
2731            end_if;
2732            show objPIDSFrames[nCurrentFrame];
2733        end_if;
2734    end;
2735
2736    #
2737    #-------------------------------------------------
2738    # Initialization function used at start up of
2739    # the PIDS application...
2740    #-------------------------------------------------
2741    #
2742    function Init;
2743        nLabValuesCreated = 0;
2744        #
2745        text_entry_allergies_selector.object_string = sTitles[0];
2746        text_entry_med_history_selector.object_string = sTitles[1];
2747        text_entry_family_history_selector.object_string = sTitles[2];
2748        text_entry_social_history_selector.object_string = sTitles[3];
2749        #
2750        drug_selector.list = drugs_list;
2751        disease_selector.list = diseases_list;
2752        text_entry_allergies_selector.list = allergies_list;
2753        text_entry_med_history_selector.list = drugs_list;
```

```
2754      text_entry_family_history_selector.list = diseases_list;
2755      text_entry_social_history_selector.list = social_list;
2756      #
2757      nDBPrefsKeyField = 1;
2758      fDBPrefsEnableSpeedEntryField = false;
2759      sDBPrefsWirelessURLField = "";
2760      sDBPrefsLocationIDField = "";
2761      sDBPrefsRotationIDField = "";
2762      sDBPrefsUserIDField = "";
2763      #
2764      if (dba_ffOpenDB(dbPidsPrefs, "dbPidsPrefs"));
2765          if (dba_ffRecordExists(dbPidsPrefs, 1));
2766              if (dba_ffReadExactRecord(dbPidsPrefs, 1) = false);
2767                  ErrorMsg("Unable to read from the PIDS preferences database.");
2768              end_if;
2769          else_if (dba_ffWriteRecord(dbPidsPrefs, 1) = false);
2770              ErrorMsg("Unable to write to the PIDS preferences database.");
2771          end_if;
2772          call dba_CloseDB(dbPidsPrefs);
2773      else;
2774          ErrorMsg("Unable to open the PIDS preferences database.");
2775      end_if;
2776      #
2777      fSpeedEntryEnabled = fDBPrefsEnableSpeedEntryField;
2778      sWirelessURL = sDBPrefsWirelessURLField;
2779      sLocationID = sDBPrefsLocationIDField;
2780      sRotationID = sDBPrefsRotationIDField;
2781      sUserID = sDBPrefsUserIDField;
2782      #
2783  end;
2784
2785  #
2786  #-------------------------------------------------
2787  # Function used to retrieve the next key id...
2788  #-------------------------------------------------
2789  #
2790  function GetInterventionKeyId as numeric;
2791      if (fUpdateMode);
2792          GetInterventionKeyId = nInterventionKeyId;
2793      else;
2794          GetInterventionKeyId = -1;
2795          if (dba_ffOpenDB(dbPids, "dbPids"));
2796              #
2797              # Retrieve the total records currently in the database...
2798              errorcode = ce_no_error;
2799              seek_end dbPids, 1;
2800              if (errorcode = ce_no_error);
2801                  errorcode = ce_no_error;
2802                  get_fields dbPids;
2803                  if (errorcode = ce_no_error);
2804                      GetInterventionKeyId = nDBPIDSKeyField + 1;
2805                  else_if (errorcode = ce_eof);
2806                      GetInterventionKeyId = 1;
2807                  else;
2808                      ErrorMsg("Unable to obtain the next key id from the PIDS database.");
2809                  end_if;
2810              else;
2811                  ErrorMsg("Unable to obtain the next key id from the PIDS database.");
2812              end_if;
2813              #
2814              dba_CloseDB(dbPids);
2815          else;
2816              ErrorMsg("Unable to open PIDS preferences database to obtain the next key
                id.");
2817          end_if;
2818      end_if;
2819  end;
2820
2821  #
2822  #---------------------------------
2823  # DB manipulation functions.
```

```
#-----------------------------------
#
function SaveXMLInterventionRecord(numeric nKey, string sSyncURL) as numeric;
    SaveXMLInterventionRecord = false;
    if (dba_ffOpenDB(dbPidsXML, "dbPidsXML"));
        #
        # Now save the XML record...
        nDBXMLPIDSKeyField = nDBPIDSKeyField;
        nDBXMLSynced = 0; # Always must be 0 (not synced)
        sDBXMLIntervention = BuildXMLRecord(sSyncURL);
        #
        SaveXMLInterventionRecord = dba_ffWriteRecord(dbPidsXML, nKey);
        if (not SaveXMLInterventionRecord);
            ErrorMsg("Unable to save the intervention to the PIDS XML database.");
        end_if;
        #
        dba_CloseDB(dbPidsXML);
    else;
        ErrorMsg("Unable to open PIDS interventions XML database.");
    end_if;
end;

function SaveInterventionRecord(numeric nKey, string sSyncURL) as numeric;
    variables;
        numeric nResult;
    end;
    SaveInterventionRecord = false;
    if (dba_ffOpenDB(dbPids, "dbPids"));
        SaveInterventionRecord = dba_ffWriteRecord(dbPids, nKey);
        if (not SaveInterventionRecord);
            dba_CloseDB(dbPids);
            ErrorMsg("Unable to save the intervention to the PIDS database.");
        else;
            dba_CloseDB(dbPids);
            #
            # Save was successfull so proceed to save the corresponding
            # XML record...
            SaveInterventionRecord = SaveXMLInterventionRecord(nKey, sSyncURL);
            if (not SaveInterventionRecord);
                #
                # Since we now require both databases to have identical records
                # (keys, etc.) we must "back out" the original saved intervention
                # since the XML formatted record was not successfully saved.
                nResult = dba_ffRemoveRecord(dbPids, nKey);
            end_if;
        end_if;
    else;
        ErrorMsg("Unable to open PIDS interventions database.");
    end_if;
end;


function DeleteXMLInterventionRecord(numeric nKey) as numeric;
    DeleteXMLInterventionRecord = false;
    if (dba_ffOpenDB(dbPidsXML, "dbPidsXML"));
        DeleteXMLInterventionRecord = dba_ffRemoveRecord(dbPidsXML, nKey);
        if (not (DeleteXMLInterventionRecord));
            ErrorMsg("Unable to delete the intervention.");
        end_if;
        #
        dba_CloseDB(dbPidsXML);
    else;
        ErrorMsg("Unable to open PIDS interventions XML database.");
    end_if;
end;

function DeleteInterventionRecord(numeric nKey) as numeric;
    DeleteInterventionRecord = false;
    if (dba_ffOpenDB(dbPids, "dbPids"));
        DeleteInterventionRecord = dba_ffRemoveRecord(dbPids, nKey);
        if (not (DeleteInterventionRecord));
```

```
2895            dba_CloseDB(dbPids);
2896            ErrorMsg("Unable to delete the intervention.");
2897        else;
2898            dba_CloseDB(dbPids);
2899            # Delete successfull so proceed to delete the corresponding
2900            # XML record...
2901            DeleteInterventionRecord = DeleteXMLInterventionRecord(nKey);
2902        end_if;
2903
2904    else;
2905        ErrorMsg("Unable to open PIDS interventions database.");
2906    end_if;
2907 end;
2908
2909 function SavePreferences as numeric;
2910     SavePreferences = false;
2911     if (dba_ffOpenDB(dbPidsPrefs, "dbPidsPrefs"));
2912         #
2913         nDBPrefsKeyField = 1;
2914         fDBPrefsEnableSpeedEntryField = cbPrefsSpeedEntryCheckbox.checkbox;
2915         sDBPrefsWirelessURLField = txPrefsWirelessSyncURL.display;
2916         sDBPrefsLocationIDField = txPrefsLocationID.display;
2917         sDBPrefsRotationIDField = txPrefsRotationID.display;
2918         sDBPrefsUserIDField = txPrefsUserID.display;
2919         #
2920         SavePreferences = dba_ffWriteRecord(dbPidsPrefs, nDBPrefsKeyField);
2921         if (SavePreferences);
2922             fSpeedEntryEnabled = fDBPrefsEnableSpeedEntryField;
2923             sWirelessURL = sDBPrefsWirelessURLField;
2924             sLocationID = sDBPrefsLocationIDField;
2925             sRotationID = sDBPrefsRotationIDField;
2926             sUserID = sDBPrefsUserIDField;
2927             #
2928             # NOTE may need to delete this code out later...
2929             txLocationID.display = sLocationID;
2930             txRotationID.display = sRotationID;
2931             txUserID.display = sLocationID;
2932         else;
2933             ErrorMsg("Unable to save to PIDS preferences database.");
2934         end_if;
2935         #
2936         dba_CloseDB(dbPidsPrefs);
2937     else;
2938         ErrorMsg("Unable to open PIDS preferences database.");
2939     end_if;
2940 end;
2941
2942 #
2943 #-------------------------------------------------
2944 # Pop-up selector functions...
2945 #-------------------------------------------------
2946 #
2947 function Compare(string sString1, string sString2) as numeric;
2948     variables;
2949         numeric nChar1;
2950         numeric nChar2;
2951         numeric nIndex;
2952         numeric nLength;
2953         numeric fFinished;
2954     end;
2955     #
2956     #  -1  indicates that sString1 < sString2
2957     #   0  indicates that they are equal
2958     #   1  indicates that sString1 > sString2
2959     #
2960     if (length(sString1) < length(sString2));
2961         nLength = length(sString1);
2962     else_if (length(sString1) > length(sString2));
2963         nLength = length(sString2);
2964     else;
2965         nLength = length(sString1);
```

```
2966        end_if;
2967        #
2968        fFinished = false;
2969        nIndex = 0;
2970        Compare = 0; # Always start by assuming they're equal...
2971        while ((nIndex < nLength) and (not fFinished));
2972            nChar1 = asc(mid(sString1, nIndex, 1));
2973            nChar2 = asc(mid(sString2, nIndex, 1));
2974            if (nChar1 < nChar2);
2975                Compare = -1;
2976                fFinished = true;
2977            else_if (nChar1 > nChar2);
2978                Compare = 1;
2979                fFinished = true;
2980            else;
2981                nIndex = nIndex + 1;
2982            end_if;
2983        end_while;
2984        #
2985        if (Compare = 0);
2986            if (length(sString1) < length(sString2));
2987                Compare = -1;
2988            else;
2989                Compare = 1;
2990            end_if;
2991        end_if;
2992    end;
2993
2994    function BinarySearch(string saList[], string sItem, numeric nFirst, numeric nLast) as
    numeric;
2995        variables;
2996            numeric nMid;
2997        end;
2998        if (nFirst > nLast);
2999            BinarySearch = -1;
3000        else;
3001            nMid = Trunc((nFirst + nLast) / 2);
3002            if (sItem = saList[nMid]);
3003                BinarySearch = nMid;
3004            else_if (Compare(sItem, saList[nMid]) = -1); # sItem < saList[nMid]...
3005                BinarySearch = BinarySearch(saList, sItem, nFirst, (nMid - 1));
3006            else;
3007                BinarySearch = BinarySearch(saList, sItem, (nMid + 1), nLast);
3008            end_if;
3009        end_if;
3010    end;
3011
3012    function BestFitSearch(string saList[], string sItem, numeric nFirst, numeric nLast) as
    numeric;
3013        variables;
3014            numeric nMid;
3015        end;
3016        if (nFirst > nLast);
3017            BestFitSearch = nFirst; # Only difference between binary and best fit is this
                line...
3018        else;
3019            nMid = Trunc((nFirst + nLast) / 2);
3020            if (sItem = saList[nMid]);
3021                BestFitSearch = nMid;
3022            else_if (Compare(sItem, saList[nMid]) = -1); # sItem < saList[nMid]...
3023                BestFitSearch = BestFitSearch(saList, sItem, nFirst, (nMid - 1));
3024            else;
3025                BestFitSearch = BestFitSearch(saList, sItem, (nMid + 1), nLast);
3026            end_if;
3027        end_if;
3028    end;
3029
3030    function ShowDrugSelector;
3031        variables;
3032            string sBestFitString;
3033            numeric nSelectedIndex;
```

```
3034        end;
3035        if (length(txPrescribedDrug.display) > 0);
3036            sBestFitString = txPrescribedDrug.display;
3037            if (length(sBestFitString) >= 1);
3038                sBestFitString = upper(left(sBestFitString, 1)) + mid(sBestFitString, 1,
                    (length(sBestFitString) - 1));
3039            end_if;
3040            nSelectedIndex = BestFitSearch(drugs_list, sBestFitString, 0, (nDrugsListSize -
                1));
3041            drug_selector.selected = nSelectedIndex;
3042        end_if;
3043        show drug_selector;
3044    end;
3045
3046    function DrugSelected;
3047        variables;
3048            string sItemSelected;
3049        end;
3050        hide drug_selector;
3051        get invoker, sItemSelected;
3052        txPrescribedDrug.display = sItemSelected;
3053    end;
3054
3055    function ShowDiseaseSelector;
3056        variables;
3057            string sBestFitString;
3058            numeric nSelectedIndex;
3059        end;
3060        if (length(txDisease.display) > 0);
3061            # Must start at element 2 since item 0 and 1 are "Unknown" and "Other" which
3062            # aren't in alphabetical order and would make this BestFirstSearch() call fail...
3063            sBestFitString = txDisease.display;
3064            if (length(sBestFitString) >= 1);
3065                sBestFitString = upper(left(sBestFitString, 1)) + mid(sBestFitString, 1,
                    (length(sBestFitString) - 1));
3066            end_if;
3067            nSelectedIndex = BestFitSearch(diseases_list, sBestFitString, 2,
                (nDiseasesListSize - 1));
3068            disease_selector.selected = nSelectedIndex;
3069        end_if;
3070        show disease_selector;
3071    end;
3072
3073    function DiseaseSelected;
3074        variables;
3075            string sItemSelected;
3076        end;
3077        hide disease_selector;
3078        get invoker, sItemSelected;
3079        txDisease.display = sItemSelected;
3080        if (sItemSelected = "Asthma");
3081            call HideFrame;
3082            show DSW_asthma_frame;
3083        end_if;
3084    end;
3085
3086    function ShowTextEntrySelector;
3087        variables;
3088            string sSelectorType;
3089        end;
3090        sSelectorType = invoker.object_string;
3091        if (sSelectorType = sTitles[0]);
3092            show text_entry_allergies_selector;
3093        else_if (sSelectorType = sTitles[1]);
3094            show text_entry_med_history_selector;
3095        else_if (sSelectorType = sTitles[2]);
3096            show text_entry_family_history_selector;
3097        else_if (sSelectorType = sTitles[3]);
3098            show text_entry_social_history_selector;
3099        end_if;
3100    end;
```

```
3101
3102 function ShowPatientIDSelector;
3103     show patient_id_selector;
3104 end;
3105
3106 function GenericItemSelected;
3107     variables;
3108         string sSelectorType;
3109         string sItemSelected;
3110     end;
3111     sSelectorType = invoker.object_string;
3112     if (sSelectorType = sTitles[0]);
3113         hide text_entry_allergies_selector;
3114     else_if (sSelectorType = sTitles[1]);
3115         hide text_entry_med_history_selector;
3116     else_if (sSelectorType = sTitles[2]);
3117         hide text_entry_family_history_selector;
3118     else_if (sSelectorType = sTitles[3]);
3119         hide text_entry_social_history_selector;
3120     end_if;
3121     #
3122     get invoker, sItemSelected;
3123     if (length(txEntry.display) > 0);
3124         txEntry.display = txEntry.display + char(10) + sItemSelected;
3125     else;
3126         txEntry.display = sItemSelected;
3127     end_if;
3128 end;
3129
3130 #
3131 #-------------------------------------------------
3132 # Lab Values functions...
3133 #-------------------------------------------------
3134 #
3135 function LabValueTypeSelected;
3136     variables;
3137         numeric nSelectedItem;
3138     end;
3139     lab_value_types_label.display = invoker.display;
3140     #
3141     nSelectedItem = invoker.selected;
3142     # Blood Pressure
3143     if (nSelectedItem = 1);
3144         type_descriptions_selector.list = blood_pressure_list;
3145     # Respiratory
3146     else_if (nSelectedItem = 2);
3147         type_descriptions_selector.list = respiratory_value_list;
3148     # Blood Sugar
3149     else_if (nSelectedItem = 3);
3150         type_descriptions_selector.list = blood_sugar_values;
3151     # Lipids
3152     else_if (nSelectedItem = 4);
3153         type_descriptions_selector.list = lipid_values_list;
3154     # Osteoporosis
3155     else_if (nSelectedItem = 5);
3156         type_descriptions_selector.list = osteoporosis_values_list;
3157     # Liver Function Tests
3158     else_if (nSelectedItem = 6);
3159         type_descriptions_selector.list = liver_function_tests_list;
3160     # Electrolytes
3161     else_if (nSelectedItem = 7);
3162         type_descriptions_selector.list = electrolytes_list;
3163     # Anticoagulation
3164     else_if (nSelectedItem = 8);
3165         type_descriptions_selector.list = anticoagulation_values_list;
3166     # Renal Function
3167     else_if (nSelectedItem = 9);
3168         type_descriptions_selector.list = renal_function_list;
3169     # Hematology
3170     else_if (nSelectedItem = 10);
3171         type_descriptions_selector.list = hematology_list;
```

```
3172        end_if;
3173        #
3174        type_descriptions_selector.selected = 0;
3175        type_descriptions_label.display = type_descriptions_selector.display;
3176    end;
3177
3178    function TypeDescriptionsSelected;
3179        txtLabValue.focused = 1; # Set the focus on the value text field...
3180        type_descriptions_label.display = invoker.display;
3181    end;
3182
3183    function UnitsOfMeasureSelected;
3184        units_of_measure_label.display = invoker.display;
3185    end;
3186
3187    function ShowLabValueTypesSelector;
3188        show lab_value_types_selector;
3189    end;
3190
3191    function ShowTypeDescriptionsSelector;
3192        show type_descriptions_selector;
3193    end;
3194
3195    function ShowUnitsOfMeasureSelector;
3196        show units_of_measure_selector;
3197    end;
3198
3199    function EnterLabValues;
3200        call HideFrame;
3201        show add_lab_values_frame;
3202        lab_value_types_selector.selected = 0;
3203        lab_value_types_label.display = lab_value_types_selector.display;
3204        type_descriptions_selector.selected = 0;
3205        type_descriptions_label.display = type_descriptions_selector.display;
3206        units_of_measure_selector.selected = 0;
3207        units_of_measure_label.display = units_of_measure_selector.display;
3208        completed_lab_values_selector.selected = -1;
3209    end;
3210
3211    function AddLabValue;
3212        variables;
3213            string sUnitsOfMeasure;
3214            string sLabValueDescription;
3215        end;
3216        #
3217        if ((nLabValuesCreated < 20) and (lab_value_types_selector.selected > 0) and
            (type_descriptions_selector.selected > 0) and (length(txtLabValue.display) > 0));
3218            if (units_of_measure_selector.selected > 0);
3219                sUnitsOfMeasure = " " + units_of_measure_label.display;
3220            else;
3221                sUnitsOfMeasure = "";
3222            end_if;
3223            sLabValueDescription = type_descriptions_label.display + " " + txtLabValue.display
                + sUnitsOfMeasure;
3226            completed_lab_values[nLabValuesCreated] = sLabValueDescription;
3225            #
3226            nLabValuesCreated = nLabValuesCreated + 1;
3227            #
3228            lab_value_types_selector.selected = 0;
3229            lab_value_types_label.display = lab_value_types_selector.display;
3230            #
3231            type_descriptions_selector.selected = 0;
3232            type_descriptions_label.display = type_descriptions_selector.display;
3233            #
3234            units_of_measure_selector.selected = 0;
3235            units_of_measure_label.display = units_of_measure_selector.display;
3236            #
3237            completed_lab_values_selector.selected = -1;
3238            #
3239            show completed_lab_values_selector;
3240            #
```

```
3241            txtLabValue.display = "";
3242        end_if;
3243    end;
3244
3245    function ClearLabValuesList;
3246        variables;
3247            numeric nIndex;
3248        end;
3249        for nIndex = 0, nIndex < 20;
3250            completed_lab_values[nIndex] = "";
3251        next nIndex + 1;
3252        nLabValuesCreated = 0;
3253    end;
3254
3255    function AcceptLabValues;
3256        hide add_lab_values_frame;
3257        call ShowFrame;
3258        txtLabValues.display = GetLabValuesData(txtLabValues.display);
3259        txtObjective.display = GetLabValuesData(txObjective.display);
3260        call ClearLabValuesList;
3261    end;
3262
3263    function CancelLabValues;
3264        hide add_lab_values_frame;
3265        call ShowFrame;
3266    end;
3267
3268    function DeleteLabValue;
3269        variables;
3270            numeric nIndex;
3271            numeric nLoopIndex;
3272        end;
3273        #
3274        nIndex = completed_lab_values_selector.selected;
3275        if ((nLabValuesCreated > 0) and (nIndex < nLabValuesCreated));
3276            nLabValuesCreated = nLabValuesCreated - 1;
3277            #
3278            for nLoopIndex = nIndex, nLoopIndex < nLabValuesCreated;
3279                completed_lab_values[nLoopIndex] = completed_lab_values[nIndex + 1];
3280            next nLoopIndex + 1;
3281            #
3282            for nLoopIndex = nLabValuesCreated, nLoopIndex < 20;
3283                completed_lab_values[nLoopIndex] = "";
3284            next nLoopIndex + 1;
3285            #
3286            completed_lab_values_selector.selected = -1;
3287            #
3288            show completed_lab_values_selector;
3289        else;
3290            completed_lab_values_selector.selected = -1;
3291        end_if;
3292    end;
3293
3294    function PickUserSelectedInterventionDate;
3295        variables;
3296            string sPreviousDate;
3297        end;
3298        sPreviousDate = intervention_date_button.display;
3299        intervention_date_button.display = GetInterventionDate(sPreviousDate);
3300    end;
3301
3302
```

```
#
# PIDS_sync.cpk
#
################################################################################
#
# PIDS (tm)
# Pharmacy Intervention Documentation System
# Copyright © 2002  Integrated Software Systems, L.L.C.
#
################################################################################
#
# Version:  1.0
# Date:     11/24/2001
# Author:   Todd Viegut & Brad Tice
#
#
# Suite of functions to perform wireless synchronization..
#
include "PIDS_error_handling.cpk";
include "PIDS_db_files.cpk";
include "dba_DBAccess.cpk";     # include generic db access functions

#
variables;
    string sInputString;
end;

#
#-----------------------------------------------
#   SOCKET CONNECTION DEFINITIONS
#       FOR INTERNET ACCESS
#-----------------------------------------------
#
file sync_socket;           # file object for triggering open socket
end;

#
# Read data from the socket
function sync_socket;
    variables;
        numeric selection;
    end;
    errorcode = 0;
    get sync_socket, sInputString;
    if errorcode = 0;
        if bytesread <> 0;
            #txResponse.display = txResponse.display + sInputString;
        end_if;
    else;
        selection = message_box(1,"PIDS Error", "Wireless sync error occurred during read:
         " + cne_geterror, "","Ok","");
    end_if;
end;

variables;
    string DONT_ENCODE =
    "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789_.-*";
    string HEX_DIGITS = "0123456789ABCDEF";
end;

function Hex(numeric nDecimal) as string;
    variables;
        numeric nQuotient;
        numeric nRemainder;
    end;
    nQuotient = trunc(nDecimal / 16);
    nRemainder = nDecimal % 16;
    #
    if (nQuotient > 0);
        Hex = Hex(nQuotient) + mid(HEX_DIGITS, nRemainder, 1);
    else;
```

```
3372            Hex = mid(HEX_DIGITS, nRemainder, 1);
3373        end_if;
3374    end;
3375
3376    function URLEncode(string sInput) as string;
3377        variables;
3378            string sHex;
3379            string sOutput;
3380            string sChar;
3381            numeric nLoop;
3382        end;
3383        for nLoop = 0, nLoop < length(sInput);
3384            sChar = mid(sInput, nLoop, 1);
3385            #
3386            # If the current char is not in the DONT_ENCODE
3387            # string then proceed to encode it...
3388            if (not (find(sChar, DONT_ENCODE, 0) >= 0));
3389                if (sChar = " ");
3390                    sOutput = sOutput + "+";
3391                else;
3392                    sHex = Hex(asc(sChar));
3393                    if (length(sHex) = 1) or (length(sHex) = 3);
3394                        sHex = "0" + sHex;
3395                    end_if;
3396                    if (length(sHex) = 4);
3397                        sOutput = sOutput + "%" + left(sHex, 2);
3398                        sOutput = sOutput + "%" + right(sHex, 2);
3399                    else;
3400                        sOutput = sOutput + "%" + sHex;
3401                    end_if;
3402                end_if;
3403            else;
3404                sOutput = sOutput + sChar;
3405            end_if;
3406        next nLoop + 1;
3407        URLEncode = sOutput;
3408    end;
3409
3410    function BuildPostRequest(string sRelSyncURL, string sURLEncodedPostString) as string;
3411        BuildPostRequest = "POST " + sRelSyncURL + " HTTP/1.0" + char(13) + char(10)
3412                           + "Content-type: application/x-www-form-urlencoded" + char(13) +
                               char(10)
3413                           + "Content-length: " + string(length(sURLEncodedPostString), "#") +
                               char(13) + char(10) + char(13) + char(10)
3414                           + sURLEncodedPostString
3415                           + char(13) + char(10);
3416    end;
3417
3418    function BuildURLEncodedPostString() as string;
3419        variables;
3420            string sPostString;
3421        end;
3422        #
3423        sPostString = sPostString + "pids_1=" + string(nDBPIDSKeyField, "#") + "&";
3424        sPostString = sPostString + "pids_2=" + string(nDBSynced, "#") + "&";
3425        #
3426        sPostString = sPostString + "pids_3=" + URLEncode(sDBRotationID) + "&";
3427        sPostString = sPostString + "pids_4=" + URLEncode(sDBLocationID) + "&";
3428        sPostString = sPostString + "pids_5=" + URLEncode(sDBInterventionDate) + "&";
3429        sPostString = sPostString + "pids_6=" + URLEncode(sDBUserID) + "&";
3430        sPostString = sPostString + "pids_7=" + URLEncode(sDBPrescribedDrug) + "&";
3431        sPostString = sPostString + "pids_8=" + URLEncode(sDBDisease) + "&";
3432        sPostString = sPostString + "pids_9=" + URLEncode(sDBPrescriber) + "&";
3433        #
3434        sPostString = sPostString + "pids_10=" + string(nDBDrugRelatedProblemField, "#") +
            "&";
3435        sPostString = sPostString + "pids_11=" + string(nDBRXTypeField, "#") + "&";
3436        sPostString = sPostString + "pids_12=" + string(nDBPatientRiskField, "#") + "&";
3437        sPostString = sPostString + "pids_13=" + string(nDBProblemTypeField, "#") + "&";
3438        sPostString = sPostString + "pids_14=" + string(nDBRecommendationField, "#") + "&";
3439        sPostString = sPostString + "pids_15=" + string(nDBActivityField, "#") + "&";
```

```
3440        sPostString = sPostString + "pids_16=" + string(nDBResultField, "#") + "&";
3441        sPostString = sPostString + "pids_17=" + string(nDBTimeField, "#") + "&";
3442        #
3443        sPostString = sPostString + "pids_18=" + URLEncode(sDBPatientIDField) + "&";
3444        sPostString = sPostString + "pids_19=" + URLEncode(sDBAllergiesField) + "&";
3445        sPostString = sPostString + "pids_20=" + URLEncode(sDBMedicalHistoryField) + "&";
3446        sPostString = sPostString + "pids_21=" + URLEncode(sDBFamilyHistoryField) + "&";
3447        sPostString = sPostString + "pids_22=" + URLEncode(sDBSocialHistoryField) + "&";
3448        #
3449        sPostString = sPostString + "pids_23=" + URLEncode(sDBSOAPSubjectiveField) + "&";
3450        sPostString = sPostString + "pids_24=" + URLEncode(sDBSOAPObjectiveField) + "&";
3451        sPostString = sPostString + "pids_25=" + URLEncode(sDBSOAPAssessmentField) + "&";
3452        sPostString = sPostString + "pids_26=" + URLEncode(sDBSOAPPlanField) + "&";
3453        sPostString = sPostString + "pids_27=" + URLEncode(sDBSOAPFollowUpField) + "&";
3454        #
3455        sPostString = sPostString + "pids_28=" + URLEncode(sDBLabValues) + "&";
3456        sPostString = sPostString + "pids_29=" + string(nDBDOIScore, "#");
3457        #
3458        BuildURLEncodedPostString = sPostString;
3459    end;
3460
3461    function BuildXMLRecord(string sSyncURL) as string;
3462        variables;
3463            string sXMLRecord;
3464        end;
3465        #
3466        sXMLRecord = sXMLRecord + "<?xml version='1.0'?>";
3467        sXMLRecord = sXMLRecord + "<!DOCTYPE intervention_set PUBLIC '-//Integrated Software
            Systems, LLC//Intervention Sets//EN'
            'http://www.pidsware.com/dtds/intervention_set_1_0.dtd'>";
3468        sXMLRecord = sXMLRecord + "<intervention_set sync_url='" + sSyncURL + "'>";
3469        sXMLRecord = sXMLRecord + "<intervention>";
3470        sXMLRecord = sXMLRecord + "<key url_param_name='pids_1'>" + string(nDBPIDSKeyField,
            "#") + "</key>";
3471        sXMLRecord = sXMLRecord + "<sync_flag url_param_name='pids_2'>" + string(nDBSynced,
            "#") + "</sync_flag>";
3472        #
3473        sXMLRecord = sXMLRecord + "<rotation_id url_param_name='pids_3'>" + sDBRotationID +
            "</rotation_id>";
3474        sXMLRecord = sXMLRecord + "<location_id url_param_name='pids_4'>" + sDBLocationID +
            "</location_id>";
3475        sXMLRecord = sXMLRecord + "<intervention_date url_param_name='pids_5'>" +
            sDBInterventionDate + "</intervention_date>";
3476        sXMLRecord = sXMLRecord + "<user_id url_param_name='pids_6'>" + sDBUserID +
            "</user_id>";
3477        sXMLRecord = sXMLRecord + "<prescribed_drug url_param_name='pids_7'>" +
            sDBPrescribedDrug + "</prescribed_drug>";
3478        sXMLRecord = sXMLRecord + "<disease url_param_name='pids_8'>" + sDBDisease +
            "</disease>";
3479        sXMLRecord = sXMLRecord + "<prescriber url_param_name='pids_9'>" + sDBPrescriber +
            "</prescriber>";
3480        #
3481        sXMLRecord = sXMLRecord + "<drug_related_problem url_param_name='pids_10'>" +
            string(nDBDrugRelatedProblemField, "#") + "</drug_related_problem>";
3482        sXMLRecord = sXMLRecord + "<rx_type url_param_name='pids_11'>" +
            string(nDBRXTypeField, "#") + "</rx_type>";
3483        sXMLRecord = sXMLRecord + "<patient_risk url_param_name='pids_12'>" +
            string(nDBPatientRiskField, "#") + "</patient_risk>";
3484        sXMLRecord = sXMLRecord + "<problem_type url_param_name='pids_13'>" +
            string(nDBProblemTypeField, "#") + "</problem_type>";
3485        sXMLRecord = sXMLRecord + "<recommendation url_param_name='pids_14'>" +
            string(nDBRecommendationField, "#") + "</recommendation>";
3486        sXMLRecord = sXMLRecord + "<activity url_param_name='pids_15'>" +
            string(nDBActivityField, "#") + "</activity>";
3487        sXMLRecord = sXMLRecord + "<result url_param_name='pids_16'>" + string(nDBResultField,
            "#") + "</result>";
3488        sXMLRecord = sXMLRecord + "<time url_param_name='pids_17'>" + string(nDBTimeField,
            "#") + "</time>";
3489        #
3490        sXMLRecord = sXMLRecord + "<patient_id url_param_name='pids_18'>" + sDBPatientIDField
            + "</patient_id>";
```

```
3491        sXMLRecord = sXMLRecord + "<allergies url_param_name='pids_19'>" + sDBAllergiesField +
            "</allergies>";
3492        sXMLRecord = sXMLRecord + "<medical_history url_param_name='pids_20'>" +
            sDBMedicalHistoryField + "</medical_history>";
3493        sXMLRecord = sXMLRecord + "<family_history url_param_name='pids_21'>" +
            sDBFamilyHistoryField + "</family_history>";
3494        sXMLRecord = sXMLRecord + "<social_history url_param_name='pids_22'>" +
            sDBSocialHistoryField + "</social_history>";
3495        #
3496        sXMLRecord = sXMLRecord + "<soap_subjective url_param_name='pids_23'>" +
            sDBSOAPSubjectiveField + "</soap_subjective>";
3497        sXMLRecord = sXMLRecord + "<soap_objective url_param_name='pids_24'>" +
            sDBSOAPObjectiveField + "</soap_objective>";
3498        sXMLRecord = sXMLRecord + "<soap_assessment url_param_name='pids_25'>" +
            sDBSOAPAssessmentField + "</soap_assessment>";
3499        sXMLRecord = sXMLRecord + "<soap_plan url_param_name='pids_26'>" + sDBSOAPPlanField +
            "</soap_plan>";
3500        sXMLRecord = sXMLRecord + "<soap_follow_up url_param_name='pids_27'>" +
            sDBSOAPFollowUpField + "</soap_follow_up>";
3501        #
3502        sXMLRecord = sXMLRecord + "<lab_values url_param_name='pids_28'>" + sDBLabValues +
            "</lab_values>";
3503        sXMLRecord = sXMLRecord + "<doi_score url_param_name='pids_29'>" + string(nDBDOIScore,
            "#") + "</doi_score>";
3504        sXMLRecord = sXMLRecord + "</intervention>";
3505        sXMLRecord = sXMLRecord + "</intervention_set>";
3506        #
3507        BuildXMLRecord = sXMLRecord;
3508    end;
3509
3510    #
3511    # Write data to the socket
3512    function SyncIntervention(string sSyncURL, string sRequest) as numeric;
3513        close sync_socket;
3514        #
3515        open sync_socket, sSyncURL;
3516        sInputString = sRequest;
3517        #
3518        errorcode = ce_no_error;
3519        put sync_socket, sInputString, length(sInputString);
3520        #
3521        #close sync_socket;
3522        #
3523        SyncIntervention = errorcode;
3524    end;
3525
3526    function ExtractRelativeSyncURL(string sURL) as string;
3527        variables;
3528            numeric nIndex;
3529        end;
3530        ExtractRelativeSyncURL = "";
3531        nIndex = find("//", sURL, 0);
3532        if (nIndex > -1);
3533            nIndex = find("/", sURL, (nIndex + 2));
3534            ExtractRelativeSyncURL = Mid(sURL, nIndex, (length(sURL) - nIndex));
3535        end_if;
3536    end;
3537
3538    function SyncAllInterventions(string sSyncURL);
3539        variables;
3540            string sRelativeSyncURL;
3541            string sURLEncodedString;
3542            string sRequest;
3543            numeric nLocalErrorCode;
3544        end;
3545        #
3546        # Retrieve the total records currently in the database...
3547        if (dba_ffOpenDB(dbPids, "dbPids"));
3548            sRelativeSyncURL = ExtractRelativeSyncURL(sSyncURL);
3549            errorcode = ce_no_error;
3550            seek_start dbPids, 0;
```

```
3551            while (errorcode = ce_no_error);
3552                get_fields dbPids;
3553                if errorcode = ce_no_error;
3554                    sURLEncodedString = BuildURLEncodedPostString();
3555                    sRequest = BuildPostRequest(sRelativeSyncURL, sURLEncodedString);
3556                    nLocalErrorCode = SyncIntervention(sSyncURL, sRequest);
3557                    errorcode = nLocalErrorCode;
3558                    if errorcode <> ce_no_error;
3559                        ErrorMsg("Wireless sync error occurred.");
3560                    end_if;
3561                end_if;
3562            end_while;
3563            call dba_CloseDB(dbPids);
3564        else;
3565            ErrorMsg("Unable to open PIDS intervention database.");
3566        end_if;
3567    end;
3568
3569
```

```
#
# PIDS_wizards.cpk
#
################################################################################
#
# PIDS (tm)
# Pharmacy Intervention Documentation System
# Copyright © 2002  Integrated Software Systems, L.L.C.
#
################################################################################
#
# Version:  1.0
# Date:     1/10/2002
# Author:   Todd Viegut & Brad Tice
#
#
# Disease State Wizards:
#


#
#=============================================================
#   A S T H M A   D I S E A S E   S T A T E   W I Z A R D
#=============================================================
#
#
#-------------------------------------------------
# Invoked functions associated with the
# DSW Asthma Disclaimer screen...
#-------------------------------------------------
#
function DSWAcceptAsthmaDisclaimer;
    hide DSW_asthma_frame;
    show DSW_asthma_age_frame;
end;

function DSWCancelAsthmaDisclaimer;
    hide DSW_asthma_frame;
    call ShowFrame;
end;

#
#-------------------------------------------------
# Invoked functions associated with the
# DSW Asthma Age screen...
#-------------------------------------------------
#
function DSWBackToAsthmaDisclaimer;
    hide DSW_asthma_age_frame;
    show DSW_asthma_frame;
end;

function DSWNextToAsthmaEval;
    hide DSW_asthma_age_frame;
    show DSW_asthma_eval_frame;
end;

function DSWAsthmaAgeSelected;
#   hide DSW_asthma_age_frame;
    if (invoker.selected > 0);
        show DSW_asthma_eval_frame;
        #
        days_with_symptoms_selector.selected = 0;
        nights_with_symptoms_selector.selected = 0;
        expiratory_flow_selector.selected = 0;
        expiratory_volume_selector.selected = 0;
        #
        days_with_symptoms_label.display = days_with_symptoms_selector.display;
        nights_with_symptoms_label.display = nights_with_symptoms_selector.display;
        expiratory_flow_label.display = expiratory_flow_selector.display;
        expiratory_volume_label.display = expiratory_volume_selector.display;
```

```
3641      end_if;
3642  end;
3643
3644  #
3645  #-------------------------------------------------
3646  # Invoked functions associated with the
3647  # DSW Asthma Evaluation screen...
3648  #-------------------------------------------------
3649  #
3650  function DSWBackToAsthmaAge;
3651      hide DSW_asthma_eval_frame;
3652      show DSW_asthma_age_frame;
3653  end;
3654
3655  function DSWNextToAsthmaAge;
3656      hide DSW_asthma_eval_frame;
3657      show DSW_asthma_age_frame;
3658  end;
3659
3660  function DSWShowDaysWithSymptomsSelector;
3661      show days_with_symptoms_selector;
3662  end;
3663
3664  function DSWShowNightsWithSymptomsSelector;
3665      show nights_with_symptoms_selector;
3666  end;
3667
3668  function DSWShowExpiratoryFlowSelector;
3669      show expiratory_flow_selector;
3670  end;
3671
3672  function DSWShowExpiratoryVolumeSelector;
3673      show expiratory_volume_selector;
3674  end;
3675
3676  function DSWDaysWithSymptomsSelected;
3677      if (invoker.selected > 0);
3678          days_with_symptoms_label.display = invoker.display;
3679      end_if;
3680  end;
3681
3682  function DSWNightsWithSymptomsSelected;
3683      if (invoker.selected > 0);
3684          nights_with_symptoms_label.display = invoker.display;
3685      end_if;
3686  end;
3687
3688  function DSWExpiratoryFlowSelected;
3689      if (invoker.selected > 0);
3690          expiratory_flow_label.display = invoker.display;
3691      end_if;
3692  end;
3693
3694  function DSWExpiratoryVolumeSelected;
3695      if (invoker.selected > 0);
3696          expiratory_volume_label.display = invoker.display;
3697      end_if;
3698  end;
3699  #
3700  #============================================================
3701  #  L I P I D S   D I S E A S E   S T A T E   W I Z A R D
3702  #============================================================
3703  #
3704  #
3705
3706
```

```
1  package com.pidsware.pids.conduit;
2
3  import palm.conduit.*;
4  import java.io.*;
5  import java.net.URLEncoder;
6  /**
7   *
8   * InterventionRecord.java
9   *
10  *#########################################################################
11  *
12  * PIDS (tm)
13  * Pharmacy Intervention Documentation System
14  * PIDS (tm) Conduit for performing Palm HotSync®s
15  * Copyright © 2002  Integrated Software Systems, L.L.C.
16  *
17  *#########################################################################
18  *
19  *
20  * Subclassed record object for an intervention record on the HH.
21  *
22  *
23  * @version 1.0
24  * @date    08.10.2001
25  * @author: Todd Viegut & Brad Tice
26  */
27 public class InterventionRecord extends AbstractRecord {
28      private double interventionKey;
29      private double synced;
30      private String rotationID = null;
31      private String locationID = null;
32      private String interventionDate = null;
33      private String userID = null;
34      private String prescribedDrug = null;
35      private String disease = null;
36      private String prescriber = null;
37      private double drugRelatedProblem;
38      private double rxType;
39      private double patientRisk;
40      private double problemType;
41      private double recommendation;
42      private double activity;
43      private double result;
44      private double interventionDuration;
45      private String patientID = null;
46      private String allergies = null;
47      private String medicalHistory = null;
48      private String familyHistory = null;
49      private String socialHistory = null;
50      private String subjective = null;
51      private String objective = null;
52      private String assessment = null;
53      private String plan = null;
54      private String followUp = null;
55      private String labValues = null;
56      private double doiScore;
57  public double getActivity() {
58      return this.activity;
59  }
60  public String getAllergies() {
61      return this.allergies;
62  }
63  public String getAssessment() {
64      return this.assessment;
65  }
66  public String getDisease() {
67      return this.disease;
68  }
69  public double getDOIScore() {
70      return doiScore;
71  }
```

```
72  public double getDrugRelatedProblem() {
73      return this.drugRelatedProblem;
74  }
75  public String getFamilyHistory() {
76      return this.familyHistory;
77  }
78  public String getFollowUp() {
79      return this.followUp;
80  }
81  public String getInterventionDate() {
82      return this.interventionDate;
83  }
84  public double getInterventionDuration() {
85      return this.interventionDuration;
86  }
87  public double getInterventionKey() {
88      return this.interventionKey;
89  }
90  public String getLabValues() {
91      return this.labValues;
92  }
93  public String getLocationID() {
94      return this.locationID;
95  }
96  public String getMedicalHistory() {
97      return this.medicalHistory;
98  }
99  public String getObjective() {
100     return this.objective;
101 }
102 public String getPatientID() {
103     return this.patientID;
104 }
105 public double getPatientRisk() {
106     return this.patientRisk;
107 }
108 public String getPlan() {
109     return this.plan;
110 }
111 public String getPrescribedDrug() {
112     return this.prescribedDrug;
113 }
114 public String getPrescriber() {
115     return this.prescriber;
116 }
117 public double getProblemType() {
118     return this.problemType;
119 }
120 public double getRecommendation() {
121     return this.recommendation;
122 }
123 public double getResult() {
124     return this.result;
125 }
126 /**
127  * Insert the method's description here.
128  * Creation date: (12/29/2001 11:25:36 AM)
129  * @return java.lang.String
130  */
131 public String getRotationID() {
132     return this.rotationID;
133 }
134 public double getRXType() {
135     return this.rxType;
136 }
137 public String getSocialHistory() {
138     return this.socialHistory;
139 }
140 public String getSubjective() {
141     return this.subjective;
142 }
```

```
143  public double getSynced() {
144      return this.synced;
145  }
146  public String getUserID() {
147      return this.userID;
148  }
149  /** Returns a null String if the parameter s is empty (contains no characters).
150   */
151  String nullIfEmpty(String s) {
152      if ("".equals(s)) {
153          return null;
154      } else {
155          return s;
156      }
157  }
158  /** Converts a DataInputStream representing the body of the hand-held device
159   *  record into Record object members.  The format of the DataInputStream
160   *  must be understood and translated by the developer.
161   */
162  public void readData(DataInputStream in) throws IOException {
163      //
164      //================================================
165      //     I N T E R V E N T I O N   H E A D E R
166      //================================================
167      //
168
169      // PIDS intervention key
170      this.interventionKey = in.readDouble();
171
172      // Lab Values
173      this.synced = in.readDouble();
174
175      // Rotation ID
176      this.rotationID = readCString(in);
177
178      // Site ID
179      this.locationID = readCString(in);
180
181      // Intervention Date
182      this.interventionDate = readCString(in);
183
184      // Student ID
185      this.userID = readCString(in);
186
187      // Prescribed Drug
188      this.prescribedDrug = readCString(in);
189
190      // Disease
191      this.disease = readCString(in);
192
193      // Prescriber
194      this.prescriber = readCString(in);
195
196      //
197      //================================================
198      //          P A T I E N T   I N F O
199      //================================================
200      //
201
202      // Drug Related Problem
203      this.drugRelatedProblem = in.readDouble();
204
205
206      // RX Type
207      this.rxType = in.readDouble();
208
209
210      // Patient Risk
211      this.patientRisk = in.readDouble();
212
213
```

```
214     // Problem Type
215     this.problemType = in.readDouble();
216
217
218     // Recommendation
219     this.recommendation = in.readDouble();
220
221
222     // Activity
223     this.activity = in.readDouble();
224
225
226     // Result
227     this.result = in.readDouble();
228
229
230     // InterventionDuration
231     this.interventionDuration = in.readDouble();
232
233     //
234     //=================================================
235     //          P A T I E N T   I N F O
236     //=================================================
237     //
238
239     // Patient ID
240     this.patientID = readCString(in);
241
242     // Allergies notes
243     this.allergies = readCString(in);
244
245     // Medical History notes
246     this.medicalHistory = readCString(in);
247
248     // Family History notes
249     this.familyHistory = readCString(in);
250
251     // Social History notes
252     this.socialHistory = readCString(in);
253
254     //
255     //=================================================
256     //              S O A P   N O T E S
257     //=================================================
258     //
259
260     // Suojective notes
261     this.subjective = readCString(in);
262
263     // Objective notes
264     this.objective = readCString(in);
265
266     // Assessment notes
267     this.assessment = readCString(in);
268
269     // Plan notes
270     this.plan = readCString(in);
271
272     // Follow Up notes
273     this.followUp = readCString(in);
274
275     //
276     //=================================================
277     //              L A B   V A L U E S
278     //-------------------------------------------------
279     //
280
281     // Lab Values
282     this.labValues = readCString(in);
283
284     //
```

```
285     //=================================================
286     //                D O I   S C O R E
287     //=================================================
288     //
289
290     // DOI Score
291     this.doiScore = in.readDouble();
292
293     //
294     //=============================================================================
295     //=============================================================================
296     //=============================================================================
297     //=============================================================================
298     //=============================================================================
299     //=============================================================================
300     //=============================================================================
301     //=============================================================================
302     //
303 }
304 public void setActivity(double value) {
305     this.activity = value;
306 }
307 public void setAllergies(String value) {
308     this.allergies = nullIfEmpty(value);
309 }
310 public void setAssessment(String value) {
311     this.assessment = nullIfEmpty(value);
312 }
313 public void setDisease(String value) {
314     this.disease = nullIfEmpty(value);
315 }
316 public void setDOIScore(double value) {
317     this.doiScore = value;
318 }
319 public void setDrugRelatedProblem(double value) {
320     this.drugRelatedProblem = value;
321 }
322 public void setFamilyHistory(String value) {
323     this.familyHistory = nullIfEmpty(value);
324 }
325 public void setFollowUp(String value) {
326     this.followUp = nullIfEmpty(value);
327 }
328 public void setInterventionDate(String value) {
329     this.interventionDate = nullIfEmpty(value);
330 }
331 public void setInterventionDuration(double value) {
332     this.interventionDuration = value;
333 }
334 public void setInterventionKey(double value) {
335     this.interventionKey = value;
336 }
337 public void setLabValues(String value) {
338     this.labValues = nullIfEmpty(value);
339 }
340 public void setLocationID(String value) {
341     this.locationID = nullIfEmpty(value);
342 }
343 public void setMedicalHistory(String value) {
344     this.medicalHistory = nullIfEmpty(value);
345 }
346 public void setObjective(String value) {
347     this.objective = nullIfEmpty(value);
348 }
349 public void setPatientID(String value) {
350     this.patientID = nullIfEmpty(value);
351 }
352 public void setPatientRisk(double value) {
353     this.patientRisk = value;
354 }
355 public void setPlan(String value) {
```

```
356         this.plan = nullIfEmpty(value);
357     }
358     public void setPrescribedDrug(String value) {
359         this.prescribedDrug = nullIfEmpty(value);
360     }
361     public void setPrescriber(String value) {
362         this.prescriber = nullIfEmpty(value);
363     }
364     public void setProblemType(double value) {
365         this.problemType = value;
366     }
367     public void setRecommendation(double value) {
368         this.recommendation = value;
369     }
370     public void setResult(double value) {
371         this.result = value;
372     }
373     public void setRotationID(String value) {
374         this.rotationID = nullIfEmpty(value);
375     }
376     public void setRXType(double value) {
377         this.rxType = value;
378     }
379     public void setSocialHistory(String value) {
380         this.socialHistory = nullIfEmpty(value);
381     }
382     public void setSubjective(String value) {
383         this.subjective = nullIfEmpty(value);
384     }
385     public void setSynced(double value) {
386         this.synced = value;
387     }
388     public void setUserID(String value) {
389         this.userID = nullIfEmpty(value);
390     }
391     static String stringWithoutCarriageReturns(String aString) {
392         StringBuffer sb = new StringBuffer();
393         int i, c;
394         char ch;
395
396         for (i = 0, c = aString.length(); i < c; i++) {
397             ch = aString.charAt(i);
398             if (ch == '\r' && (i + 1) < c && aString.charAt(i + 1) == '\n')
399                 continue;
400             else
401                 sb.append(ch);
402         }
403         return sb.toString();
404     }
405     public String toFormattedString() {
406         return ("Intervention record: {\r\n" +
407                     "  key: " + getInterventionKey() + "\r\n" +
408                     "  synced: " + getSynced() + "\r\n" +
409                     "  rotation ID: " + getRotationID() + "\r\n" +
410                     "  site ID: " + getLocationID() + "\r\n" +
411                     "  intervention date: " + getInterventionDate() + "\r\n" +
412                     "  student ID: " + getUserID() + "\r\n" +
413                     "  prescribed drug: " + getPrescribedDrug() + "\r\n" +
414                     "  disease: " + getDisease() + "\r\n" +
415                     "  prescriber: " + getPrescriber() + "\r\n" +
416                     "  drug related problem: " + getDrugRelatedProblem() + "\r\n" +
417                     "  RX type: " + getRXType() + "\r\n" +
418                     "  patient risk: " + getPatientRisk() + "\r\n" +
419                     "  problem type: " + getProblemType() + "\r\n" +
420                     "  recommendation: " + getRecommendation() + "\r\n" +
421                     "  activity: " + getActivity() + "\r\n" +
422                     "  result: " + getResult() + "\r\n" +
423                     "  intervention duration: " + getInterventionDuration() + "\r\n" +

424                     "  patient ID: " + getPatientID() + "\r\n" +
425                     "  allergies: " + getAllergies() + "\r\n" +
```

```
                "  medical history: " + getMedicalHistory() + "\r\n" +
                "  family history: " + getFamilyHistory() + "\r\n" +
                "  social history: " + getSocialHistory() + "\r\n" +
                "  SOAP NOTES: " + "\r\n" +
                "------------------------------------------------------------ + "\r\n" +
                "  subjective: " + getSubjective() + "\r\n" +
                "  objective: " + getObjective() + "\r\n" +
                "  assessment: " + getAssessment() + "\r\n" +
                "  plan: " + getPlan() + "\r\n" +
                "  follow up: " + getFollowUp() + "\r\n\n" +
                "------------------------------------------------------------ + "\r\n" +
                "  lab values: " + getLabValues() + "\r\n" +
                "  DOI score: " + getDOIScore() + "\r\n" +
                "}\r\n" +
                super.toFormattedString());
}
public String toPostString() {
    StringBuffer postString = new StringBuffer();
    //
    postString
        .append("pids_1=")
        .append(URLEncoder.encode(Double.toString(getInterventionKey())))
        .append("&");
    postString
        .append("pids_2=")
        .append(URLEncoder.encode(Double.toString(getSynced())))
        .append("&");
    //
    postString.append("pids_3=").append(URLEncoder.encode(getRotationID())).append(
        "&");
    postString.append("pids_4=").append(URLEncoder.encode(getLocationID())).append(
        "&");
    postString
        .append("pids_5=")
        .append(URLEncoder.encode(getInterventionDate()))
        .append("&");
    postString.append("pids_6=").append(URLEncoder.encode(getUserID())).append("&");
    postString
        .append("pids_7=")
        .append(URLEncoder.encode(getPrescribedDrug()))
        .append("&");
    postString.append("pids_8=").append(URLEncoder.encode(getDisease())).append(
        "&");
    postString.append("pids_9=").append(URLEncoder.encode(getPrescriber())).append(
        "&");
    //
    postString
        .append("pids_10=")
        .append(URLEncoder.encode(Double.toString(getDrugRelatedProblem())))
        .append("&");
    postString
        .append("pids_11=")
        .append(URLEncoder.encode(Double.toString(getRXType())))
        .append("&");
    postString
        .append("pids_12=")
        .append(URLEncoder.encode(Double.toString(getPatientRisk())))
        .append("&");
    postString
        .append("pids_13=")
        .append(URLEncoder.encode(Double.toString(getProblemType())))
        .append("&");
    postString
        .append("pids_14=")
        .append(URLEncoder.encode(Double.toString(getRecommendation())))
        .append("&");
    postString
        .append("pids_15=")
        .append(URLEncoder.encode(Double.toString(getActivity())))
        .append("&");
    postString
```

```
497             .append("pids_16=")
498             .append(URLEncoder.encode(Double.toString(getResult())))
499             .append("&");
500         postString
501             .append("pids_17=")
502             .append(URLEncoder.encode(Double.toString(getInterventionDuration())))
503             .append("&");
504         //
505         postString.append("pids_18=").append(URLEncoder.encode(getPatientID())).append(
506             "&");
507         postString.append("pids_19=").append(URLEncoder.encode(getAllergies())).append(
508             "&");
509         postString
510             .append("pids_20=")
511             .append(URLEncoder.encode(getMedicalHistory()))
512             .append("&");
513         postString
514             .append("pids_21=")
515             .append(URLEncoder.encode(getFamilyHistory()))
516             .append("&");
517         postString
518             .append("pids_22=")
519             .append(URLEncoder.encode(getSocialHistory()))
520             .append("&");
521         //
522         postString.append("pids_23=").append(
523             URLEncoder.encode(getSubjective())).append(
524             "&");
525         postString.append("pids_24=").append(URLEncoder.encode(getObjective())).append(
526             "&");
527         postString.append("pids_25=").append(
528             URLEncoder.encode(getAssessment())).append(
529             "&");
530         postString.append("pids_26=").append(URLEncoder.encode(getPlan())).append("&");
531         postString.append("pids_27=").append(URLEncoder.encode(getFollowUp()))).append(
532             "&");
533         //
534         postString.append("pids_28=").append(URLEncoder.encode(getLabValues())).append(
535             "&");
536         postString.append("pids_29=").append(
537             URLEncoder.encode(Double.toString(getDOIScore())));
538         //
539         return postString.toString();
540     }
541     public String toString() {
542         return "Intervention record:   <rotation id="
543             + getRotationID()
544             + ">   <site id="
545             + getLocationID()
546             + ">   <patient id="
547             + getPatientID()
548             + ">   <prescribed drug="
549             + getPrescribedDrug()
550             + ">  "
551             + super.toString();
552     }
553     public void writeData(DataOutputStream out) throws IOException {
554         /*
555         //
556         //================================================
557         //     I N T E R V E N T I O N   H E A D E R
558         //================================================
559         //
560
561         // PIDS intervention key
562         out.writeDouble(this.interventionKey);
563
564         // Synced
565         out.writeDouble(this.synced);
566
567         // Rotation ID
```

```
568     writeCString(out, stringWithoutCarriageReturns(this.rotationID));
569
570     // Site ID
571     writeCString(out, stringWithoutCarriageReturns(this.locationID));
572
573     // Intervention Date
574     writeCString(out, stringWithoutCarriageReturns(this.interventionDate));
575
576     // Student ID
577     writeCString(out, stringWithoutCarriageReturns(this.userID));
578
579     // Prescribed Drug
580     writeCString(out, stringWithoutCarriageReturns(this.prescribedDrug));
581
582     // Disease
583     writeCString(out, stringWithoutCarriageReturns(this.disease));
584
585     // Prescriber
586     writeCString(out, stringWithoutCarriageReturns(this.prescriber));
587
588     //
589     //================================================
590     //      I N T E R V E N T I O N   I N F O
591     //================================================
592     //
593
594     // Drug Related Problem
595     out.writeDouble(this.drugRelatedProblem);
596
597     // RX Type
598     out.writeDouble(this.rxType);
599
600     // Patient Risk
601     out.writeDouble(this.patientRisk);
602
603     // Problem Type
604     out.writeDouble(this.problemType);
605
606     // Recommendation
607     out.writeDouble(this.recommendation);
608
609     // Activity
610     out.writeDouble(this.activity);
611
612     // Result
613     out.writeDouble(this.result);
614
615     // InterventionDuration
616     out.writeDouble(this.interventionDuration);
617
618     //
619     //================================================
620     //          P A T I E N T   I N F O
621     //================================================
622     //
623
624     // Patient ID
625     writeCString(out, stringWithoutCarriageReturns(this.patientID));
626
627     // Allergies notes
628     writeCString(out, stringWithoutCarriageReturns(this.allergies));
629
630     // Medical History notes
631     writeCString(out, stringWithoutCarriageReturns(this.medicalHistory));
632
633     // Family History notes
634     writeCString(out, stringWithoutCarriageReturns(this.familyHistory));
635
636     // Social History notes
637     writeCString(out, stringWithoutCarriageReturns(this.socialHistory));
638
```

```
639     //
640     //================================================
641     //              S O A P   N O T E S
642     //================================================
643     //
644
645     // Subjective notes
646     writeCString(out, stringWithoutCarriageReturns(this.subjective));
647
648     // Objective notes
649     writeCString(out, stringWithoutCarriageReturns(this.objective));
650
651     // Assessment notes
652     writeCString(out, stringWithoutCarriageReturns(this.assessment));
653
654     // Plan notes
655     writeCString(out, stringWithoutCarriageReturns(this.plan));
656
657     // Follow Up notes
658     writeCString(out, stringWithoutCarriageReturns(this.followUp));
659
660     //
661     //================================================
662     //              L A B   V A L U E S
663     //================================================
664     //
665
666     // Lab Values
667     writeCString(out, stringWithoutCarriageReturns(this.labValues));
668
669     //
670     //================================================
671     //              D O I   S C O R E
672     //================================================
673     //
674
675     // DOI Score
676     out.writeDouble(this.doiScore);
677
678     */
679
680     //
681     //================================================================================
682     //================================================================================
683     //================================================================================
684     //================================================================================
685     //================================================================================
686     //================================================================================
687     //================================================================================
688     //================================================================================
689     //
690
691     //
692     //================================================
693     //     I N T E R V E N T I O N   H E A D E R
694     //================================================
695     //
696
697     // PIDS intervention key
698     out.writeDouble(this.interventionKey);
699
700     // Synced
701     out.writeDouble(this.synced);
702
703     // Rotation ID
704     if (this.rotationID != null) {
705         out.write(this.rotationID.getBytes());
706         out.write(0);
707     }
708
709     // Site ID
```

```
710     if (this.locationID != null) {
711         out.write(this.locationID.getBytes());
712         out.write(0);
713     }
714
715     // Intervention Date
716     if (this.interventionDate != null) {
717         out.write(this.interventionDate.getBytes());
718         out.write(0);
719     }
720
721     // Student ID
722     if (this.userID != null) {
723         out.write(this.userID.getBytes());
724         out.write(0);
725     }
726
727     // Prescribed Drug
728     if (this.prescribedDrug != null) {
729         out.write(this.prescribedDrug.getBytes());
730         out.write(0);
731     }
732
733     // Disease
734     if (this.disease != null) {
735         out.write(this.disease.getBytes());
736         out.write(0);
737     }
738
739     // Prescriber
740     if (this.prescriber != null) {
741         out.write(this.prescriber.getBytes());
742         out.write(0);
743     }
744
745     //
746     //================================================
747     //      I N T E R V E N T I O N   I N F O
748     //================================================
749     //
750
751     // Drug Related Problem
752     out.writeDouble(this.drugRelatedProblem);
753
754     // RX Type
755     out.writeDouble(this.rxType);
756
757     // Patient Risk
758     out.writeDouble(this.patientRisk);
759
760     // Problem Type
761     out.writeDouble(this.problemType);
762
763     // Recommendation
764     out.writeDouble(this.recommendation);
765
766     // Activity
767     out.writeDouble(this.activity);
768
769     // Result
770     out.writeDouble(this.result);
771
772     // InterventionDuration
773     out.writeDouble(this.interventionDuration);
774
775     //
776     //================================================
777     //          P A T I E N T   I N F O
778     //================================================
779     //
780
```

```
781     // Patient ID
782     if (this.patientID != null) {
783         out.write(this.patientID.getBytes());
784         out.write(0);
785     }
786
787     // Allergies notes
788     if (this.allergies != null) {
789         out.write(this.allergies.getBytes());
790         out.write(0);
791     }
792
793     // Medical History notes
794     if (this.medicalHistory != null) {
795         out.write(this.medicalHistory.getBytes());
796         out.write(0);
797     }
798
799     // Family History notes
800     if (this.familyHistory != null) {
801         out.write(this.familyHistory.getBytes());
802         out.write(0);
803     }
804
805     // Social History notes
806     if (this.socialHistory != null) {
807         out.write(this.socialHistory.getBytes());
808         out.write(0);
809     }
810
811     //
812     //================================================
813     //              S O A P   N O T E S
814     //================================================
815     //
816
817     // Subjective notes
818     if (this.subjective != null) {
819         out.write(this.subjective.getBytes());
820         out.write(0);
821     }
822
823     // Objective notes
824     if (this.objective != null) {
825         out.write(this.objective.getBytes());
826         out.write(0);
827     }
828
829     // Assessment notes
830     if (this.assessment != null) {
831         out.write(this.assessment.getBytes());
832         out.write(0);
833     }
834
835     // Plan notes
836     if (this.plan != null) {
837         out.write(this.plan.getBytes());
838         out.write(0);
839     }
840
841     // Follow Up notes
842     if (this.followUp != null) {
843         out.write(this.followUp.getBytes());
844         out.write(0);
845     }
846
847     //
848     //================================================
849     //              L A B   V A L U E S
850     //================================================
851     //
```

```
852
853     // Lab Values
854
855     if (this.labValues != null) {
856         out.write(this.labValues.getBytes());
857         out.write(0);
858     }
859
860     //
861     //================================================
862     //                D O I   S C O R E
863     //================================================
864     //
865
866     // DOI Score
867     out.writeDouble(this.doiScore);
868 }
869 }
870
871
872
873
874
875
876
877
878
879
880 package com.pidsware.pids.conduit;
881
882 import java.util.*;
883
884 /**
885  *
886  * PIDSConduitBundle.java
887  *
888  *#########################################################################
889  *
890  * PIDS (tm)
891  * Pharmacy Intervention Documentation System
892  * PIDS (tm) Conduit for performing Palm HotSync®
893  * Copyright © 2002  Integrated Software Systems, L.L.C.
894  *
895  *#########################################################################
896  *
897  *
898  * Class providing localization for PIDS Conduit resources.
899  *
900  *
901  * @version 1.0
902  * @date    08.10.2001
903  * @author: Todd Viegut & Brad Tice
904  */
905
906 public class PIDSConduitBundle extends ListResourceBundle {
907     //
908     public static final char COPYRIGHT = ((char) 169);
909     public static final char REGISTERED = ((char) 174);
910     public static final char TM = ((char) 174); // This need to be changed to the actual
        char value...
911     //
912     static final Object[][] contents =
913     {
914         {"CONDUIT_NAME", "PIDS" + TM + " Conduit"},
915         {"pids_conduit_props", "pids.props"},
916         {"BAD_URL_ERROR", "PIDS" + TM + " Conduit Error:  Bad URL: "},
917         {"SYNC_ERROR", "PIDS" + TM + " Conduit Error:  Synchronization failed"},
918         {"RW_ERROR", "PIDS" + TM + " Conduit Error:  Failed to successfully sync the data
        to the URL specified"},
919         {"FILE_NOT_FOUND_ERROR", "PIDS" + TM + " Conduit Error:  URL Not Found: "},
920         {"OK_PIDS_CONDUIT", "OK PIDS" + TM + " Conduit"},
```

```
        {"PIDS_CONDUIT", "PIDS" + TM + " Conduit"},
        {"DIR_DELIMITER", "\\"},
        {"PIDS_Conduit_Do_Nothing", "PIDS" + TM + " Conduit - sync set to Do Nothing"},
        {"debug", "true"},
        {"logfile", "PIDScond.log"}
    },
public Object[][] getContents() {
    return contents;
}
}
```

```
package com.pidsware.pids.conduit;

import java.net.URL;
import java.net.HttpURLConnection;
import java.net.URLConnection;
import java.net.MalformedURLException;
import java.io.InputStreamReader;
import java.io.IOException;
import java.io.BufferedReader;
import java.io.CharArrayReader;
import java.io.Reader;

import org.xml.sax.*;
import com.ibm.xml.parsers.NonValidatingDOMParser;

/**
 *
 * DOMInterventionURLBuilder.java
 *
 *###########################################################################
 *
 * PIDS (tm)
 * Pharmacy Intervention Documentation System
 * PIDS (tm) Conduit for performing Palm HotSync®s
 * Copyright © 2002  Integrated Software Systems, L.L.C.
 *
 *###########################################################################
 *
 *
 * This class is a non-validating DOM parser which takes in an
 * XML formatted intervention, parses it, and dynamically builds
 * a URL encoded name/value parameters list that gets sent as a
 * POST request to the designated sync URL provided by the user
 * on the HH.
 *
 *
 * @version 1.0
 * @date    02.04.2002
 * @author: Todd Viegut & Brad Tice
 */

import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.io.UnsupportedEncodingException;

import org.w3c.dom.Attr;
import org.w3c.dom.Document;
import org.w3c.dom.NamedNodeMap;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.w3c.dom.Element;
```

```
992
993  public class DOMInterventionURLBuilder {
994      /** Default parser name. */
995      private static final String DEFAULT_PARSER_NAME =
         "com.ibm.xml.parsers.NonValidatingDOMParser";
996
997      /** Buffer for URL building. */
998      private StringBuffer urlPostString = new StringBuffer();
999
1000     private String syncURL;
1001
1002     private NonValidatingDOMParser parser;
1003 public DOMInterventionURLBuilder() throws ClassNotFoundException, InstantiationException,
    IllegalAccessException {
1004     this(DEFAULT_PARSER_NAME);
1005 }
1006 public DOMInterventionURLBuilder(String parserName) throws ClassNotFoundException,
    InstantiationException, IllegalAccessException {
1007     //
1008     this.parser = (NonValidatingDOMParser) Class.forName(parserName).newInstance();
1009 }
1010 public static void main(String argv[]) {
1011     String parserName = DEFAULT_PARSER_NAME;
1012     //
1013     try {
1014         DOMInterventionURLBuilder urlBuilder =
1015             new DOMInterventionURLBuilder(parserName);
1016         //
1017         urlBuilder.parse("intervention_set.xml");
1018         //
1019         System.out.println("Redirect URL:    " + urlBuilder.retrieveSyncURL());
1020         System.out.println("Post String:     " + urlBuilder.retrievePostString());
1021         System.out.println(
1022             urlBuilder.testPostRequest(
1023                 urlBuilder.retrievePostString(),
1024                 urlBuilder.retrieveSyncURL()));
1025         //
1026         urlBuilder.parse("intervention_set_1_0.xml");
1027         //
1028         System.out.println("Redirect URL:    " + urlBuilder.retrieveSyncURL());
1029         System.out.println("Post String:     " + urlBuilder.retrievePostString());
1030         System.out.println(
1031             urlBuilder.testPostRequest(
1032                 urlBuilder.retrievePostString(),
1033                 urlBuilder.retrieveSyncURL()));
1034     } catch (Exception e) {
1035         e.printStackTrace();
1036     }
1037 }
1038 public void parse(Reader reader) {
1039     try {
1040         parser.reset();
1041         //
1042         this.syncURL = null;
1043         this.urlPostString = new StringBuffer();
1044         //
1045         InputSource inputSource = new InputSource(reader);
1046         //
1047         parser.parse(inputSource);
1048         Document document = parser.getDocument();
1049         //
1050         this.traverse(document);
1051     } catch (Exception e) {
1052         String message = null;
1053         char[] charArray = new char[50000];
1054         try {
1055             int size = reader.read(charArray);
1056             message = new String(charArray, 0, size);
1057         } catch (IOException ioException) {
1058             message =
1059                 new String("An error occured.  Unable to retrieve the XML record.\n\n");
```

```
1060            }
1061            processError(message, e);
1062        }
1063    }
1064    public void parse(String uri) {
1065        try {
1066            parser.reset();
1067            //
1068            this.syncURL = null;
1069            this.urlPostString = new StringBuffer();
1070            //
1071            parser.parse(uri);
1072            Document document = parser.getDocument();
1073            //
1074            this.traverse(document);
1075        } catch (Exception e) {
1076            processError(uri, e);
1077        }
1078    }
1079    private static void printUsage() {
1080        System.err.println(
1081            "Usage: com.pidsware.pids.conduit.DOMInterventionURLBuilder [options] uri ...");
1082        System.err.println();
1083        System.err.println("options:");
1084        System.err.println("  -d name  Specify DOM parser wrapper by name.");
1085        System.err.println("           Default parser: " + DEFAULT_PARSER_NAME);
1086        System.err.println("  -h       This help screen.");
1087    }
1088    private void processError(String message, Throwable exception) {
1089        //
1090        EmailSender sender = new EmailSender(message, exception);
1091        sender.start();
1092        //
1093        exception.printStackTrace(System.err);
1094    }
1095    public String retrievePostString() {
1096        return this.urlPostString.toString();
1097    }
1098    public String retrieveSyncURL() {
1099        String syncURL = null;
1100        //
1101        Document interventionDoc = this.parser.getDocument();
1102        Element interventionSetRoot = interventionDoc.getDocumentElement();
1103        syncURL = interventionSetRoot.getAttribute("sync_url");
1104        //
1105        if ((syncURL == null) || ("".equals(syncURL))) {
1106            NodeList list = interventionSetRoot.getElementsByTagName("sync_url");
1107            Node node = list.item(0);
1108            if (node != null) {
1109                syncURL =
1110                    (((node.getFirstChild() != null)
1111                        && (node.getFirstChild().getNodeValue() != null))
1112                        ? node.getFirstChild().getNodeValue()
1113                        : "");
1114            }
1115
1116        }
1117        return syncURL;
1118    }
1119    private int testPostRequest(String postString, String syncURL) {
1120        int syncStatus = 0; // default is NOT successful...
1121        //
1122        HttpURLConnection connection = null;
1123        BufferedReader in = null;
1124        PrintWriter out = null;
1125        try {
1126            syncURL =
1127                (syncURL.toLowerCase().startsWith("tcp")
1128                    ? "http" + syncURL.substring(syncURL.indexOf(":"))
1129                    : syncURL);
1130            URL url = new URL(syncURL);
```

```
1131            connection = (HttpURLConnection) url.openConnection();
1132            connection.setRequestMethod("POST");
1133            connection.setRequestProperty(
1134                "Content-Type",
1135                "application/x-www-form-urlencoded");
1136            connection.setRequestProperty(
1137                "Content-length",
1138                Integer.toString(postString.length()));
1139            connection.setDoOutput(true);
1140            connection.setDoInput(true);
1141            //
1142            out = new PrintWriter(connection.getOutputStream());
1143            out.write(postString);
1144            out.flush();
1145            //
1146            in = new BufferedReader(new InputStreamReader(connection.getInputStream()));
1147            String inputLine;
1148            //
1149            if ((inputLine = in.readLine()) != null) {
1150                if ((inputLine != null) && (!"".equals(inputLine))) {
1151                    syncStatus = Integer.parseInt(inputLine);
1152                }
1153            }
1154        } catch (MalformedURLException malformedURLException) {
1155            malformedURLException.printStackTrace();
1156        } catch (Exception e) {
1157            e.printStackTrace();
1158        } finally {
1159            try {
1160                if (in != null) {
1161                    in.close();
1162                }
1163            } catch (IOException ioException) {
1164            }
1165            //
1166            if (out != null) {
1167                out.close();
1168            }
1169            //
1170            if (connection != null) {
1171                connection.disconnect();
1172            }
1173        }
1174        //
1175        return syncStatus;
1176    }
1177    private void traverse(Node node) {
1178        //
1179        // is there anything to do?
1180        if (node == null) {
1181            return;
1182        }
1183        //
1184        NodeList children = null;
1185        //
1186        int type = node.getNodeType();
1187        switch (type) {
1188            case Node.DOCUMENT_NODE :
1189                Node d = ((Document) node).getDocumentElement();
1190                traverse(d);
1191                break;
1192            case Node.ELEMENT_NODE :
1193                NamedNodeMap attrs = node.getAttributes();
1194                if (attrs != null) {
1195                    Node attrNode = attrs.getNamedItem("url_param_name");
1196                    if (attrNode != null) {
1197                        NodeList list = attrNode.getChildNodes();
1198                        this
1199                            .urlPostString
1200                            .append(((this.urlPostString.length() > 0) ? "&" : ""))
1201                            .append(attrNode.getNodeValue())
```

```
                        .append("= ")
                        .append(
                            (((node.getFirstChild() != null)
                                && (node.getFirstChild().getNodeValue() != null))
                                ?
                                java.net.URLEncoder.encode(node.getFirstChild().getNodeVal
                                ue())
                                : ""));
                }
            }
            //
            children = node.getChildNodes();
            if (children != null) {
                int len = children.getLength();
                for (int i = 0; i < len; i++) {
                    traverse(children.item(i));
                }
            }
            break;
        case Node.ENTITY_REFERENCE_NODE :
            children = node.getChildNodes();
            if (children != null) {
                int len = children.getLength();
                for (int i = 0; i < len; i++) {
                    traverse(children.item(i));
                }
            }
            break;
        case Node.CDATA_SECTION_NODE :
        case Node.TEXT_NODE :
            break;
    }
}
}



package com.pidsware.pids.conduit;

import java.io.*;
import sun.net.smtp.*;
/**
 *
 * EmailSender.java
 *
 *#########################################################################
 *
 * PIDS (tm)
 * Pharmacy Intervention Documentation System
 * PIDS (tm) Conduit for performing Palm HotSync©s
 * Copyright © 2002  Integrated Software Systems, L.L.C.
 *
 *#########################################################################
 *
 *
 * The PIDS (tm) beta test auto error logging facility.  This class
 * is spawned as a separate thread which will email any errors that
 * occur during a Palm HotSync® with PIDS (tm).
 *
 *
 * @version 1.0
 * @date    02.08.2002
 * @author: Todd Viegut & Brad Tice
 */
```

```
1271 public class EmailSender extends Thread {
1272     private String errorMsg;
1273     private Throwable exception;
1274 public EmailSender() {
1275     super();
1276 }
1277 public EmailSender(Runnable target) {
1278     super(target);
1279 }
1280 public EmailSender(Runnable target, String name) {
1281     super(target, name);
1282 }
1283 public EmailSender(String name) {
1284     super(name);
1285 }
1286 public EmailSender(String errorMsg, Throwable exception) {
1287     super();
1288     this.errorMsg = errorMsg;
1289     this.exception = exception;
1290 }
1291 public EmailSender(ThreadGroup group, Runnable target) {
1292     super(group, target);
1293 }
1294 public EmailSender(ThreadGroup group, Runnable target, String name) {
1295     super(group, target, name);
1296 }
1297 public EmailSender(ThreadGroup group, String name) {
1298     super(group, name);
1299 }
1300 public EmailSender(Throwable exception) {
1301     super();
1302     this.exception = exception;
1303 }
1304 public static void main(String[] args) {
1305     try {
1306         throw new java.io.IOException("TESTING");
1307     } catch (java.io.IOException e) {
1308         EmailSender sender = new EmailSender("OOOPS...", e);
1309         sender.start();
1310     }
1311 }
1312 /**
1313  * Creation date: (2/8/2002 10:32:21 AM)
1314  */
1315 public void run() {
1316     try {
1317         SmtpClient smtpClient = new SmtpClient("mail.pidsware.com");
1318
1319         //assumes localhost has the SMTP
1320         smtpClient.from("viegs@aol.com");
1321         smtpClient.to("conduit_errors@pidsware.com");
1322         PrintStream out = smtpClient.startMessage();
1323         out.println("To: conduit_errors@pidsware.com");
1324         out.println("From: viegs@aol.com");
1325         out.println("Subject: Conduit error...");
1326         if (this.errorMsg != null) {
1327             out.println(this.errorMsg);
1328         }
1329         if (this.exception != null) {
1330             exception.printStackTrace(out);
1331         }
1332         smtpClient.closeServer();
1333         //
1334         System.err.println("Conduit error email sent to beta support at
            www.pidsware.com");
1335     } catch (Exception e) {
1336     }
1337 }
1338 }
1339
1340
```

```
1341
1342
1343
1344
1345
1346
1347
1348
1349 package com.pidsware.pids.conduit;
1350
1351 import java.util.*;
1352 import java.net.*;
1353 import java.io.*;
1354 import palm.conduit.*;
1355 import org.xml.sax.*;
1356 import org.xml.sax.helpers.*;
1357
1358 /**
1359  *
1360  * PIDSConduit.java
1361  *
1362  *##############################################################################
1363  *
1364  * PIDS (tm)
1365  * Pharmacy Intervention Documentation System
1366  * PIDS (tm) Conduit for performing Palm HotSync®s
1367  * Copyright © 2002  Integrated Software Systems, L.L.C.
1368  *
1369  *##############################################################################
1370  *
1371  *
1372  * MAIN SOURCE CODE FILE FOR THE PIDS(tm) CONDUIT
1373  *
1374  *
1375  * @version 1.0
1376  * @date    08.10.2001
1377  * @author: Todd Viegut & Brad Tice
1378  */
1379 public class PIDSConduit implements Conduit {
1380     /** Reference to all strings used by PIDS Conduit */
1381     private static ResourceBundle pidsConduitBundle =
        ResourceBundle.getBundle("com.pidsware.pids.conduit.PIDSConduitBundle");
1382
1383     /** Name of the conduit - hardcoded since cannot currently get it from the registry */
1384     private static final String NAME = pidsConduitBundle.getString("CONDUIT_NAME");
1385
1386     /** This string should be replaced with the URL of the prc to be copied. */
1387     private static final String PROPS_FILE =
        pidsConduitBundle.getString("pids_conduit_props");
1388
1389     /** Vector of interventions to by synced. */
1390     private Vector interventions;
1391
1392     /** Intervention reference. */
1393     private InterventionRecord intervention;
1394 /**
1395  * PIDSConduit constructor comment.
1396  */
1397 public PIDSConduit() {
1398     super();
1399 }
1400 /**
1401  * An entry point to pass configuration parameters that is called by the
1402  * HotSync Manager conduit configuration.
1403  *
1404  * @param info ConfigureConduitInfo
1405  * @returns int
1406  */
1407 public int configure(ConfigureConduitInfo info) {
1408     ConduitConfigure config = new ConduitConfigure(info, NAME);
1409     config.createDialog();
```

```
1410
1411        // Only if the data has changed do we fill the info
1412        // parameter with the new values from the configuration dlg
1413        if (config.dataChanged) {
1414            //
1415            int propsValue = SyncProperties.SYNC_DO_NOTHING; // default
1416            propsValue = config.saveState;
1417
1418            if (config.setDefault) {
1419                //
1420                info.syncPermanent = propsValue;
1421                info.syncPref = ConfigureConduitInfo.PREF_PERMANENT;
1422            }
1423            info.syncNew = propsValue;
1424        }
1425        //
1426        Log.out("ConfigureConduitInfo:  /n" + info.toString());
1427        //
1428        return 0;
1429    }
1430    /**
1431     * This method simply displays all intervention records for the database
1432     * passed in.
1433     *
1434     * @param dbName String The name of the database for which to display records.
1435     */
1436    private void displayInterventions(String dbName) throws SyncException, IOException {
1437        // Open our handheld DB
1438        int db = SyncManager.openDB(dbName, 0, SyncManager.OPEN_READ | SyncManager.OPEN_WRITE
1439        | SyncManager.OPEN_EXCLUSIVE);
1440
1441        // Find out how many intervention records there are in the database...
1442        int count = SyncManager.getDBRecordCount(db);
1443
1444        XMLInterventionRecord record = null;
1445        //
1446        // Loop over all records
1447        for (int i = 0; i < count; i++) {
1448            record = new XMLInterventionRecord();
1449            //
1450            record.setIndex(i);
1451            SyncManager.readRecordByIndex(db, record);
1452            //
1453            Log.out(record.toFormattedString());
1454        }
1455
1456        // Close DB
1457        SyncManager.closeDB(db);
1458    }
1459    /**
1460     * Starts the application.
1461     * @param args an array of command-line arguments
1462     */
1463    public static void main(String[] args) {
1464        // Insert code to start the application here.
1465        PIDSConduit conduit = new PIDSConduit();
1466    }
1467    /**
1468     * Returns a String representation of the conduit name.
1469     */
1470    public String name() {
1471        return NAME;
1472    }
1473    /**
1474     * Opens and runs a conduit.  All conduit-specific functionality should be
1475     * entered in this method.
1476     *
1477     * @param props SyncProperties
1478     */
1479    public void open(SyncProperties props) {
        //
```

```
1480        byte[] line = new byte[50000];
1481        //
1482        try {
1483            // check what type of sync is required
1484            if (props.syncType == SyncProperties.SYNC_DO_NOTHING) {
1485                //
1486                Log.out(pidsConduitBundle.getString("PIDS_Conduit_Do_Nothing"));
1487                Log.endSync();
1488                //
1489                return;
1490            }
1491
1492            //
1493            // FIRST, before we do any "syncing", we must purge all deleted interventions...
1494            purgeDeletedInterventions("dbPidsXML");
1495            purgeDeletedInterventions("dbPids");
1496
1497            //
1498            // NOW, we can proceed to sync the interventions.
1499            syncInterventions("dbPidsXML");
1500
1501            //
1502            Log.out(pidsConduitBundle.getString("OK_PIDS_CONDUIT"));
1503            Log.endSync(); // Log we are successful
1504        } catch (MalformedURLException ex) {
1505            processError(pidsConduitBundle.getString("BAD_URL_ERROR"), ex);
1506        } catch (SyncException e) {
1507            processError(pidsConduitBundle.getString("SYNC_ERROR"), e);
1508        } catch (IOException e) {
1509            processError(pidsConduitBundle.getString("RW_ERROR"), e);
1510        } catch (Exception e) {
1511            processError(pidsConduitBundle.getString("RW_ERROR"), e);
1512        }
1513    }
1514    /**
1515     * This method posts a request to synchronize a single intervention
1516     * record to the synchronization site passed in to it.
1517     *
1518     * @param postString String
1519     * @param syncURL String
1520     * @return int
1521     */
1522    private int postRequest(String postString, String syncURL) throws SyncException {
1523        int syncStatus = 0; // default is NOT successful...
1524        //
1525        HttpURLConnection connection = null;
1526        BufferedReader in = null;
1527        PrintWriter out = null;
1528        try {
1529            syncURL =
1530                (syncURL.toLowerCase().startsWith("tcp")
1531                    ? "http" + syncURL.substring(syncURL.indexOf(":"))
1532                    : syncURL);
1533            URL url = new URL(syncURL);
1534            connection = (HttpURLConnection) url.openConnection();
1535            connection.setRequestMethod("POST");
1536            connection.setRequestProperty(
1537                "Content-Type",
1538                "application/x-www-form-urlencoded");
1539            connection.setRequestProperty(
1540                "Content-length",
1541                Integer.toString(postString.length()));
1542            connection.setDoOutput(true);
1543            connection.setDoInput(true);
1544            //
1545            out = new PrintWriter(connection.getOutputStream());
1546            out.write(postString);
1547            out.flush();
1548            //
1549            in = new BufferedReader(new InputStreamReader(connection.getInputStream()));
1550            String inputLine;
```

```
1551            //
1552            if ((inputLine = in.readLine()) != null) {
1553                if ((inputLine != null) && (!"".equals(inputLine))) {
1554                    syncStatus = Integer.parseInt(inputLine);
1555                }
1556            }
1557        } catch (Exception e) {
1558            e.printStackTrace();
1559        } finally {
1560            try {
1561                if (in != null) {
1562                    in.close();
1563                }
1564            } catch (IOException ioException) {
1565            }
1566            //
1567            if (out != null) {
1568                out.close();
1569            }
1570            //
1571            if (connection != null) {
1572                connection.disconnect();
1573            }
1574        }
1575        //
1576        return syncStatus;
1577    }
1578    /**
1579     * This method processes any exception thrown during a HotSync.  It also
1580     * launches the auto error logging thread which will send an email to
1581     * conduit_errors@pidsware.com.
1582     *
1583     * @param message String
1584     * @param exception Throwable
1585     */
1586    private void processError(String message, Throwable exception) {
1587        //
1588        // log(ex.toString());
1589        //
1590        EmailSender sender = new EmailSender(message, exception);
1591        sender.start();
1592        //
1593        exception.printStackTrace();
1594        //
1595        Log.err(message + " " + exception.toString() + "\n");
1596        Log.abortSync();
1597    }
1598    /**
1599     * This method will purge any deleted records from the HH for the database named passed to
         it.
1600     *
1601     * @param dbName String
1602     */
1603    private void purgeDeletedInterventions(String dbName) throws SyncException {
1604        int db = SyncManager.openDB(dbName, 0, SyncManager.OPEN_READ | SyncManager.OPEN_WRITE
             | SyncManager.OPEN_EXCLUSIVE);
1605        SyncManager.purgeDeletedRecs(db);
1606        SyncManager.closeDB(db);
1607    }
1608    /**
1609     * This method will retrieve from the HH the url of the site to synchronize to.
1610     *
1611     * @return String
1612     * @param dbName String
1613     */
1614    private String readSyncURL(String dbName) throws SyncException, IOException {
1615        int db = -1;
1616        String url = null;
1617        try {
1618            //
1619            db =
```

```
                SyncManager.openDB(
                    dbName,
                    0,
                    SyncManager.OPEN_READ | SyncManager.OPEN_EXCLUSIVE);
            //
            PrefsRecord record = new PrefsRecord();
            //
            record.setIndex(0);
            SyncManager.readRecordByIndex(db, record);
            //
            if ((record.getURL() != null) && (!"".equals(record.getURL()))) {
                url = record.getURL();
                //
                url =
                    (url.toLowerCase().startsWith("tcp")
                        ? "http" + url.substring(url.indexOf(":"))
                        : url);
            }
        } finally {
            // Close DB
            if (db != -1) {
                SyncManager.closeDB(db);
            }
        }
        //
        return url;
    }
    /**
     * Method to synchronize the interventions from the HH to a site on the internet.
     *
     * @param dbName String The name of the database that will by synchronized.
     */
    private void syncInterventions(String dbName)
        throws SyncException, IOException {
        Vector syncedRecords = new Vector();
        //
        String syncURL = readSyncURL("dbPidsPrefs");
        //
        //=============================================================
        //      F I R S T   S Y N C   X M L   D A T A B A S E
        //=============================================================
        //
        // Open our handheld DB
        int db =
            SyncManager.openDB(
                dbName,
                0,
                SyncManager.OPEN_READ | SyncManager.OPEN_WRITE | SyncManager.OPEN_EXCLUSIVE);

        // Find out how many intervention records there are in the database...
        int count = SyncManager.getDBRecordCount(db);

        XMLInterventionRecord record = null;
        //
        DOMInterventionURLBuilder urlBuilder = null;
        //
        // Loop over all intervention records
        CharArrayReader reader = null;
        //
        for (int i = 0; i < count; i++) {
            record = new XMLInterventionRecord();
            //
            record.setIndex(i);
            SyncManager.readRecordByIndex(db, record);
            //
            if (record.getSynced() == 0.0) {
                //
                // Now use the DOM Parser to build the POST string for the url...
                char[] xmlCharArray =
                    ((record.getXMLRecord() != null)
                        ? record.getXMLRecord().toCharArray()
```

```
                    : "".toCharArray());
            reader = new CharArrayReader(xmlCharArray);
            //
            try {
                urlBuilder = new DOMInterventionURLBuilder();
                //
                urlBuilder.parse(reader);
                //
                if (postRequest(urlBuilder.retrievePostString(), syncURL) == 1) {
                    //
                    // Now write the intervention back as "synced"...
                    syncedRecords.addElement(new Integer(record.getId()));
                }
            } catch (Exception e) {
                throw new SyncException(
                    "PIDS intervention syncing failed for intervention record id "
                        + record.getId()
                        + ".  This may be the result of an invalid character entered in an
                        intervention SOAP note that the conduit is unable to process.
                        Please email beta@pidsware.com if this occurs.");
            }

        }
    }

    // Close DB
    SyncManager.closeDB(db);

    //
    //=============================================================
    //       N O W   W R I T E   R E C O R D S   B A C K
    //         F L A G I N G   I T   A S   S Y N C E D
    //=============================================================
    //
    //
    db =
        SyncManager.openDB(
            dbName,
            0,
            SyncManager.OPEN_READ | SyncManager.OPEN_WRITE | SyncManager.OPEN_EXCLUSIVE);

    // Find out how many intervention records there are in the database...
    count = SyncManager.getDBRecordCount(db);

    record = null;
    //
    // Loop over all intervention records
    Integer id = null;
    Enumeration records = syncedRecords.elements();
    //
    while (records.hasMoreElements()) {
        // for (int i = 0; i < count; i++) {
        id = (Integer) records.nextElement();
        //
        record = new XMLInterventionRecord();
        //
        record.setId(id.intValue());
        SyncManager.readRecordById(db, record);
        //
        // Now write the XML intervention back to the DB flaged as "synced"...
        record.setSynced(1.0);
        SyncManager.writeRec(db, (Record) record);
    }
    // Close DB
    SyncManager.closeDB(db);
}
}
```

```
1760
1761
1762
1763
1764
1765
1766
1767 package com.pidsware.pids.conduit;
1768
1769 import java.io.*;
1770 import java.util.*;
1771 import java.sql.*;
1772 /**
1773  *
1774  * DOMInterventionURLBuilder.java
1775  *
1776  *###########################################################################
1777  *
1778  * PIDS (tm)
1779  * Pharmacy Intervention Documentation System
1780  * PIDS (tm) Conduit for performing Palm HotSync®s
1781  * Copyright © 2002  Integrated Software Systems, L.L.C.
1782  *
1783  *###########################################################################
1784  *
1785  *
1786  * This class is a non-validating DOM parser which takes in an
1787  * XML formatted intervention, parses it, and dynamically builds
1788  * a URL encoded name/value parameters list that gets sent as a
1789  * POST request to the designated sync URL provided by the user
1790  * on the HH.
1791  *
1792  *
1793  * @version 1.0
1794  * @date    12.15.2001
1795  * @author: Todd Viegut & Brad Tice
1796  */
1797 public class ConduitDebugLog {
1798     private PrintWriter log = null;
1799     static private ConduitDebugLog instance; // The single instance
1800 /**
1801  * DebugLog constructor comment.
1802  */
1803 private ConduitDebugLog() {
1804     init();
1805 }
1806 public static synchronized ConduitDebugLog getInstance() {
1807     if (instance == null) {
1808         instance = new ConduitDebugLog();
1809     }
1810     return instance;
1811 }
1812 /**
1813  * Loads properties and initializes the instance with its values.
1814  */
1815 private void init() {
1816     InputStream is = getClass().getResourceAsStream("PIDScond.properties");
1817     Properties dbProps = new Properties();
1818     try {
1819         dbProps.load(is);
1820         String logFile = dbProps.getProperty("logfile", "PIDScond.log");
1821         try {
1822             log = new PrintWriter(new FileWriter(logFile, true), true);
1823         } catch (IOException e) {
1824             System.err.println("Can't open the log file: " + logFile);
1825             log = new PrintWriter(System.err);
1826         }
1827     } catch (Exception e) {
1828         System.err.println("Can't open the log file: " + "PIDSCond.log" + "    Using
        default logging to standard err.");
1829         log = new PrintWriter(System.err);
```

```
1830        }
1831
1832    }
1833    /**
1834     * Writes a message to the log file.
1835     */
1836    public synchronized void log(Object obj, String msg) {
1837        log.println(obj.getClass().getName() + " [" + new java.util.Date() + "]:    " + msg);
1838    }
1839    /**
1840     * Writes a message to the log file.
1841     */
1842    public synchronized void log(Object obj, String msg, Throwable exception) {
1843        log.println(obj.getClass().getName() + " [" + new java.util.Date() + "]:    " + msg);
1844        exception.printStackTrace(log);
1845    }
1846    /**
1847     * Writes a message to the log file.
1848     */
1849    public synchronized void log(String msg) {
1850        log.println("[" + new java.util.Date() + "]: " + msg);
1851    }
1852    /**
1853     * Writes a message with an Exception to the log file.
1854     */
1855    public synchronized void log(String msg, Throwable exception) {
1856        log.println("[" + new java.util.Date() + "]: " + msg);
1857        exception.printStackTrace(log);
1858    }
1859    /**
1860     * Starts the application.
1861     * @param args an array of command-line arguments
1862     */
1863    public static void main(java.lang.String[] args) {
1864        // Insert code to start the application here
1865    }
1866    }
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877    package com.pidsware.pids.conduit;
1878
1879    import palm.conduit.*;
1880    import java.io.*;
1881    import java.net.URLEncoder;
1882    /**
1883     *
1884     * PrefsRecord.java
1885     *
1886     *###############################################################################
1887     *
1888     * PIDS (tm)
1889     * Pharmacy Intervention Documentation System
1890     * PIDS (tm) Conduit for performing Palm HotSync®s
1891     * Copyright © 2002  Integrated Software Systems, L.L.C.
1892     *
1893     *###############################################################################
1894     *
1895     *
1896     * Subclassed record object for the a preference database record on
1897     * the HH.
1898     *
1899     *
1900     * @version 1.0
```

```
1901   * @date    02.10.2002
1902   * @author: Todd Viegut & Brad Tice
1903   */
1904  public class PrefsRecord extends AbstractRecord {
1905      private double prefsKey;
1906      private double enableSpeedEntry;
1907      private String url = null;
1908      private String locationID = null;
1909      private String rotationID = null;
1910      private String userID = null;
1911  public double getEnableSpeedEntry() {
1912      return this.enableSpeedEntry;
1913  }
1914  public String getLocationID() {
1915      return this.locationID;
1916  }
1917  public double getPrefsKey() {
1918      return this.prefsKey;
1919  }
1920  public String getRotationID() {
1921      return this.rotationID;
1922  }
1923  public String getURL() {
1924      return this.url;
1925  }
1926  public String getUserID() {
1927      return this.userID;
1928  }
1929  String nullIfEmpty(String s) {
1930      if ("".equals(s)) {
1931          return null;
1932      } else {
1933          return s;
1934      }
1935  }
1936  public void readData(DataInputStream in) throws IOException {
1937      //
1938      //================================================
1939      //              P R E F E R E N C E S
1940      //================================================
1941      //
1942
1943      // PIDS intervention key
1944      this.prefsKey = in.readDouble();
1945
1946      // Lab Values
1947      this.enableSpeedEntry = in.readDouble();
1948
1949      // Wireless URL
1950      this.url = readCString(in);
1951
1952      // Location ID
1953      this.locationID = readCString(in);
1954
1955      // Rotation Date
1956      this.rotationID = readCString(in);
1957
1958      // User ID
1959      this.userID = readCString(in);
1960  }
1961  public void setLocationID(String value) {
1962      this.locationID = nullIfEmpty(value);
1963  }
1964  public void setPrefsKey(double value) {
1965      this.prefsKey = value;
1966  }
1967  public void setRotationID(String value) {
1968      this.rotationID = nullIfEmpty(value);
1969  }
1970  public void setURL(String url) {
1971      this.url = nullIfEmpty(url);
```

```
1972 }
1973 public void setUserID(String value) {
1974     this.userID = nullIfEmpty(value);
1975 }
1976 static String stringWithoutCarriageReturns(String aString) {
1977     StringBuffer sb = new StringBuffer();
1978     int i, c;
1979     char ch;
1980
1981     for (i = 0, c = aString.length(); i < c; i++) {
1982         ch = aString.charAt(i);
1983         if (ch == '\r' && (i + 1) < c && aString.charAt(i + 1) == '\n')
1984             continue;
1985         else
1986             sb.append(ch);
1987     }
1988     return sb.toString();
1989 }
1990 public String toFormattedString() {
1991     return (
1992         "Intervention record: {\r\n"
1993             + "  key: "
1994             + getPrefsKey()
1995             + "\r\n"
1996             + "  enable speed entry: "
1997             + getEnableSpeedEntry()
1998             + "\r\n"
1999             + "  url: "
2000             + getURL()
2001             + "\r\n"
2002             + "  location ID: "
2003             + getLocationID()
2004             + "\r\n"
2005             + "  rotation ID: "
2006             + getRotationID()
2007             + "\r\n"
2008             + "  user ID: "
2009             + getUserID()
2010             + "\r\n"
2011             + "}\r\n"
2012             + super.toFormattedString());
2013 }
2014 public String toString() {
2015     return "prefs record:   <enable speed entry="
2016         + getEnableSpeedEntry()
2017         + ">   <url="
2018         + getURL()
2019         + ">   <rotation id="
2020         + getRotationID()
2021         + ">   <location id="
2022         + getLocationID()
2023         + ">   <user id="
2024         + getUserID()
2025         + ">   "
2026         + super.toString();
2027 }
2028 public void writeData(DataOutputStream out) throws IOException {
2029
2030     // PIDS intervention key
2031     out.writeDouble(this.prefsKey);
2032
2033     // Synced
2034     out.writeDouble(this.enableSpeedEntry);
2035
2036     // URL
2037     if (this.url != null) {
2038         out.write(this.url.getBytes());
2039         out.write(0);
2040     }
2041
2042     // Location ID
```

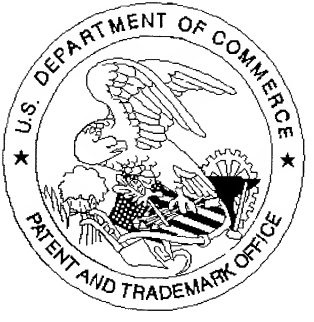
```
2043        if (this.locationID != null) {
2044            out.write(this.locationID.getBytes());
2045            out.write(0);
2046        }
2047
2048        // Rotation ID
2049        if (this.rotationID != null) {
2050            out.write(this.rotationID.getBytes());
2051            out.write(0);
2052        }
2053
2054        // User ID
2055        if (this.userID != null) {
2056            out.write(this.userID.getBytes());
2057            out.write(0);
2058        }
2059    }
2060    }
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071  package com.pidsware.pids.conduit;
2072
2073  import java.io.*;
2074  import palm.conduit.*;
2075  /**
2076   *
2077   * XMLInterventionRecord.java
2078   *
2079   *###########################################################################
2080   *
2081   * PIDS (tm)
2082   * Pharmacy Intervention Documentation System
2083   * PIDS (tm) Conduit for performing Palm HotSync®s
2084   * Copyright © 2002  Integrated Software Systems, L.L.C.
2085   *
2086   *###########################################################################
2087   *
2088   *
2089   * Subclassed record object for the an intervention stored as XML on
2090   * the HH.
2091   *
2092   *
2093   * @version 1.0
2094   * @date    02.04.2002
2095   * @author: Todd Viegut & Brad Tice
2096   */
2097  public class XMLInterventionRecord extends AbstractRecord {
2098      private double interventionKey;
2099      private double synced;
2100      private String xmlRecord = null;
2101  public double getInterventionKey() {
2102      return this.interventionKey;
2103  }
2104  public double getSynced() {
2105      return this.synced;
2106  }
2107  /** Converts and copies doPidsXML members into a DataOutputStream in preparation
2108   *  for writing to the hand-held device.
2109   *  The format of the DataOutputStream specific to the record body byte
2110   *  layout of the memo records stored on the hand-held device for the PIDS
2111   *  application.
2112   *  @param out The DataOutputStream to fill with XML record body information
2113   */
```

```
2114
2115  /** Returns the XML record.
2116   */
2117  public String getXMLRecord() {
2118      return this.xmlRecord;
2119  }
2120  String nullIfEmpty(String s) {
2121      if ("".equals(s)) {
2122          return null;
2123      } else {
2124          return s;
2125      }
2126  }
2127  public void readData(DataInputStream in) throws IOException {
2128      //
2129      //=========================================================
2130      //        I N T E R V E N T I O N   H E A D E R
2131      //=========================================================
2132      //
2133
2134      // PIDS intervention key
2135      this.interventionKey = in.readDouble();
2136
2137      // Synced flag
2138      this.synced = in.readDouble();
2139
2140      //
2141      //=========================================================
2142      //     X M L   I N T E R V E N T I O N   R E C O R D
2143      //=========================================================
2144      //
2145
2146      // PIDS Intervention record as XML
2147      this.xmlRecord = readCString(in);
2148  }
2149  public void setInterventionKey(double value) {
2150      this.interventionKey = value;
2151  }
2152  public void setSynced(double value) {
2153      this.synced = value;
2154  }
2155  public void setXMLRecord(String xmlRecord) {
2156      this.xmlRecord = xmlRecord;
2157  }
2158  static String stringWithoutCarriageReturns(String aString) {
2159      StringBuffer sb = new StringBuffer();
2160      int i, c;
2161      char ch;
2162
2163      for (i = 0, c = aString.length(); i < c; i++) {
2164          ch = aString.charAt(i);
2165          if (ch == '\r' && (i + 1) < c && aString.charAt(i + 1) == '\n')
2166              continue;
2167          else
2168              sb.append(ch);
2169      }
2170      return sb.toString();
2171  }
2172  public String toFormattedString() {
2173      StringBuffer buffer = new StringBuffer();
2174
          buffer.append("\n\n==========================================================").append("\
          n")
2175            .append("INTERVENTION").append("\n")
2176            .append("------------ ).append("\n\n")
2177            .append("Intervention Key:    ").append(getInterventionKey()).append("\n")
2178            .append("Synced Flag:         ").append(getSynced()).append("\n\n")
2179            .append("XML Record:").append("\n")
2180            .append(getXMLRecord()).append("\n\n")
2181            .append(super.toFormattedString()).append("\n")
2182            .append("------------------------------------------------------------ ).append("\n");
```

```
2183        return buffer.toString();
2184    }
2185    public String toString() {
2186        StringBuffer buffer = new StringBuffer();
2187        buffer.append("XML Intervention Record:").append("\n")
2188              .append("key:      ").append(getInterventionKey()).append("\n")
2189              .append("synced:   ").append(getSynced()).append("\n")
2190              .append("xml:      ").append(getXMLRecord()).append("\n")
2191              .append(super.toString()).append("\n");
2192        return buffer.toString();
2193    }
2194    public void writeData(DataOutputStream out) throws IOException {
2195        // XML record
2196        /*
2197        if (this.xmlRecord != null) {
2198            out.write(this.xmlRecord.getBytes());
2199            out.write(0);
2200        }
2201        */
2202
2203        // PIDS intervention key
2204        out.writeDouble(this.interventionKey);
2205
2206        // Synced
2207        out.writeDouble(this.synced);
2208
2209        // Rotation ID
2210        if (this.xmlRecord != null) {
2211            out.write(this.xmlRecord.getBytes());
2212            out.write(0);
2213        }
2214        //writeCString(out, stringWithoutCarriageReturns(this.xmlRecord));
2215    }
2216    }
2217
```

# United States Patent & Trademark Office

Office of Initial Patent Examination -- Scanning Division




Application deficiencies found during scanning:

☐ Page(s)_____ of _____ were not present for scanning. (Document title)

☐ Page(s)_____ of _____ were not present for scanning. (Document title)

***Scanned copy is best available.*** There are 60 page of specification. Rest of the pages are appendix.